commodore

COMPUTER CLUB

13 L. 2.500

La rivista degli utenti di sistemi Commodore

Mensile 25 Agosto - 25 Settembre 1984 - Anno III - N. 13 - Sped.Abb. Post. Gr. III/70 - CR Distr. MePe

SPECIALE DIDATTICA

Nuovi comandi Basic

Gestione Interrupt

File relativi

Ipermonitor L.M.

Alberi binari

Disassembler

Nuovi giochi per il VIC 20 CMB 64

NUMERO SPECIALE 100 PAGINE

systems Communications

commodore
COMPUTER CLUB

SOMMARIO

PAG.	REMarks	Vic 20	C 64	Sistemi	Generali
05	Commodore novità			•	•
09	Domande / Risposte	•	•	•	•
14	Orologio		•		
20	Il sistema "Music 64"		•		
24	Dama cinese		•		
29	Caccia al ragno	•			
35	Gli Alberi Binari di ricerca	•	•		
45	Iper Monitor per Linguaggio Macchina		•		
53	Disassembler		•		
56	Espansione dei comandi Basic		•		
59	La Gestione dell'Interrupt nel C 64		•		
62	Interrupt: applicazioni pratiche		•		
65	Linguaggio Macchina (3 puntata)		•		
71	File relativi		•		
75	Il Gioco del 15		•		
78	Space Defender		•		
80	Labirinto	•			
82	Sintetizzatore Sonoro		•		
85	Bomber		•		
87	L'informatica a scuola: Telematica, Robotica, Burotica				•

Direttore responsabile: Michele Di Pisa
Redattore capo: Alessandro de Simone
Segretaria di redazione: Maura Ceccaroli
Foto di copertina: Franco Vignati
Impaginazione/Illustrazioni: Francesco Amatori, Renato Caruso
Direzione, redazione: V.le Famagosta, 75 - 20142 Milano - Tel. (02) 8467348
Pubblicità: Mirco Croce (coordinatore) Michela Prandini, Giorgio Ruffoni, Claudio Tidone, Villa Claudio
V.le Famagosta, 75 - 20142 Milano - Tel. (02) 8467348/9/40
Prezzo e abbonamenti: Prezzo per una copia L. 2.500
Arretrati il doppio. Abbonamento annuo (10 fascicoli) L. 22.000
Abbonamento annuo cumulativo alle riviste Computer e Commodore Computer Club (tariffa riservata agli studenti) L. 34.000. I versamenti vanno indirizzati a: Commodore Computer Club mediante assegno bancario, o utilizzando il c/c postale n. 31532203.
Stampa: La Litografica s.r.l. - Busto A.
Registrazione: Tribunale di Milano n. 370 del 2/1/1982
Sped. in abb. post. Gr. III Pubblicità inferiore al 70%.

COMMODORE PLUS/4

COMMODORE 16

COMMODORE PLUS/4

Commodore Plus/4 è il computer che amplia l'orizzonte dell'home e del personal computing. Il suo nome deriva dai quattro programmi applicativi incorporati nel computer e immediatamente disponibili al momento dell'accensione. Il software è contenuto nella memoria Rom del Commodore Plus/4, e si richiamano con la semplice pressione di un tasto.

Il software integrato del Commodore Plus/4 comprende le applicazioni più diffuse e più utilizzate nell'attività quotidiana: un programma di word processing per scrivere lettere e relazioni, un foglio elettronico per la pianificazione finanziaria, un database per la creazione e la gestione di un archivio, e un pacchetto di business graphics per visualizzare sotto forma di diagrammi e istogrammi i complessi dati memorizzati.

● FILE MANAGER permette di raccogliere, memorizzare, riordinare e richiamare le informazioni di qualsiasi genere: dagli indirizzi per una mailing list agli articoli di un inventario, agli elementi di un elenco personalizzato, alle ricette di cucina. Il programma offre la possibilità di riordinare e riorganizzare le informazioni raccolte per creare lettere personalizzate, etichette per spedizioni, rapporti e tabelle. Inoltre il File Manager è uno strumento flessibile adatto per qualsiasi applicazione personale, e non costringe chi lo usa a imparare complicate tecniche di programmazione per ottenere operazioni complesse.

● SPREADSHEET è un foglio elettronico incorporato che esegue automaticamente calcoli e ricalcoli su dati numerici soggetti a continue variazioni. E' l'ideale per preparare proiezioni finanziarie, bilanci familiari o aziendali, analisi di budget e per la pianificazione in generale. Ogni volta che un dato varia, il computer ricalcola istantaneamente tutti i dati correlati, fornendo i nuovi valori per l'intero foglio di lavoro.

● WORDPROCESSOR permette di creare lettere, relazioni, rapporti e documenti scritti di qualsiasi tipo con grande facilità. I testi possono essere memorizzati su disco e corretti più volte prima di procedere alla stampa su carta. Il programma è collegato con lo Spreadsheet e può richiamare tabelle di dati da includere in un documento scritto. Questa perfetta integrazione fra programmi incorporati è un concetto rivoluzionario nel settore degli home computer.

● GRAPHICS è anch'esso incorporato nel Commodore Plus/4 ed è integrato con il foglio elettronico. Permette di visualizzare sotto forma di grafico i dati inseriti nello spreadsheet, per realizzare un sistema completo di business graphics. 12 comandi Basic aggiuntivi rendono semplicissima la creazione di programmi grafici personalizzati, che possono essere memorizzati su nastro o su disco. Con Graphics il Plus/4 è in grado di disegnare cerchi, rettangoli e complesse figure geometriche, colorarle a piacere e includerle nei testi scritti con il word processor.

I programmi incorporati possono condividere e scambiare le informazioni memorizzate dal computer. Inoltre, l'opzione di screen window permette di utilizzare due programmi contemporaneamente visualizzandoli entrambi sullo schermo.

Queste caratteristiche uniche fanno del Commodore Plus/4 la macchina ideale per l'attività professionale. Ma non solo. I computer si stanno diffondendo nel mondo della scuola, e possono essere utilizzati con profitto anche a casa. Il Plus/4 ha tutte le caratteristiche per costituire lo strumento ideale per lo studente. Il Basic esteso incorporato nella Rom offre oltre 75 comandi di programmazione, compresi quelli per gestire grafica e suono. Commodore Plus/4 è il computer ideale per imparare a sviluppare la tecnica di programmazione in linguaggio Basic.

In ambiente domestico, le possibilità di utilizzo del Plus/4 sono praticamente infinite. Grazie al software integrato, computerizzare il bilancio familiare, la corrispondenza personale e l'archiviazione di dati diventa semplicissimo. Infine, le enormi potenzialità grafiche e sonore del Plus/4 trasformano il computer nel compagno di gioco ideale.

In aggiunta al software sviluppato espressamente per questo computer, Commodore Plus/4 può utilizzare anche i programmi scritti per il Commodore 16.

CARATTERISTICHE TECNICHE

Memoria
64K Ram standard di cui 60K utilizzabili in Basic dall'utente.

ROM
32K Rom standard (compresi sistema operativo interprete Basic).

Microprocessore
7501 con clock da 0.89 a 1.76 MHz.

Video
40 colonne per 25 righe di testo.

Colori
121 colori (15 colori di base con 8 gradazioni più il nero).

Set di caratteri
Lettere maiuscole e minuscole, numeri e simboli.
Caratteri in negativo e lampeggianti.
Set di caratteri grafici Pet Commodore.

Display mode
Caratteri di testo.
Grafici in alta risoluzione e in multicolor.
Testo in schermo diviso/grafici in alta risoluzione o in multicolor.

Risoluzione grafica
320 x 200 pixel.

Suono
2 generatori di suono oppure 1 generatore di suono e un generatore di rumore bianco.
9 livelli di volume per le due voci.

Tastiera
Standard qwerty con 67 tasti.
4 tasti di controllo del cursore.
4 tasti di funzione programmati, che possono essere ridefiniti dall'utente per supportare fino a 8 funzioni prescelte.
Tasti di controllo del colore.
Tasto di HELP.
Set di caratteri maiuscolo e minuscolo.
Set di caratteri grafici.
Pulsante di Reset del sistema.
Tasto di Escape.

Input/Output
User port.
Porta seriale Commodore.
Porta per cartuccia Rom e per unità disco parallela.
2 porte per joystick.
Porta di collegamento per unità registratore C1531
Uscita monitor con segnale composito/croma/luma.
Input/output audio.
Ingresso alimentazione di rete.

Caratteristiche principali
Basic esteso versione 3.5 incorporato, con oltre 75 comandi in gradi di pilotare grafica e suono.
Monitor in linguaggio macchina incorporato, con 12 comandi disponibili.
Possibilità di creazione finestre su video.
4 programmi applicativi incorporati: File Manager, Spreadsheet, Word Processing, Graphics.

Periferiche opzionali.
SFS 481 unità a disco veloce.
C 1542 unità a disco.
C 1531 registratore a cassette.
MCS 801 stampante a matrice a 7 colori.
MPS 802 stampante a matrice.
DPS 1101 stampante a margherita 'letter quality'.
C 1520 stampante/plotter a 4 colori.
C 1703 monitor a colori.

Altre periferiche
Commodore Plus/4 funziona anche in collegamento con l'unità a disco C 1541, con la stampante a matrice C 1526 e con il monitor a colori C 1701.

Dimensioni
Altezza: 66.7 mm.
Larghezza: 422.3 mm.
Profondità: 238.1 mm.

Alimentazione di rete
220-240 volts AC, 50-60 Hz con alimentatore esterno.

Consumo
8.5 watt massimo.

Colore
Grigio antracite.

COMMODORE 16

Il nuovo Commodore 16 è un home computer che offre caratteristiche che solitamente si ritrovano soltanto in macchine appartenenti a una fascia di prezzo molto più elevata. E' stato progettato appositamente per chi si avvicina per la prima volta al mondo dell'informatica, ed è perciò semplice da usare e da programmare.

La tastiera professionale di dimensioni standard è la stessa del Vic 20 e del Commodore 64, i due home computer più diffusi in Italia. Oltre ai normali tasti di

funzione, il Commodore 16 dispone di quattro tasti per il controllo del cursore, e di un tasto di Help.

Il Commodore 16 è il computer ideale per apprendere e sviluppare la propria tecnica di programmazione. Il linguaggio Basic versione 3.5, incorporato nella memoria Rom, è stato arricchito e comprende oltre 75 comandi per la gestione completa delle capacità grafiche e sonore, e per la programmazione avanzata nel linguaggio per computer più famoso del mondo. Il tasto di Help si dimostra utilissimo in fase di sviluppo dei programmi Basic, perchè permette di evidenziare le linee contenenti gli errori di sintassi, individuando le singole istruzioni inesatte. La Commodore ha preparato una serie di corsi di programmazione basati su manuali e software di facile impiego.

La didattica è un settore in cui la Commodore opera da sempre con prodotti di alta qualità e di facile impiego, in grado di accompagnare i bambini in età scolare per tutta la durata dei loro studi. Il nuovo Commodore 16 è lo strumento ideale per questo percorso educativo, e fornisce un'occasione unica per avvicinarsi al mondo del futuro e crescere di pari passo con la tecnologia.

Il Commodore 16 è anche un superbo videogioco. Con i due ingressi per i joystick, la grafica in alta risoluzione a 121 colori e i due generatori di suono incorporati, diventa un eccezionale videogame dalle caratteristiche professionali.

CARATTERISTICHE TECNICHE

Memoria
RAM 16K standard di cui 12K utilizzabili in Basic dall'utente.
ROM 32K con sistema operativo e interprete Basic incorporati.

Microprocessore
7501 con clock da 0.89 a 1.76 MHz.

Video
40 colonne per 25 righe di testo.

Colori
121 colori (15 colori di base con 8 gradazioni più il nero).

Set di caratteri
Lettere maiuscole e minuscole, numeri e simboli.
Caratteri in negativo e lampeggianti.
Set di caratteri grafici Pet Commodore.

Display mode
Caratteri di testo.
Grafici in alta risoluzione e in multicolor.
Testo in schermo diviso/grafici in alta risoluzione o in multicolor.

Risoluzione grafica
320 x 200 pixel.

Suono
2 generatori di suono oppure 1 generatore di suono e un generatore di rumore bianco.
9 livelli di volume per le due voci.

Tastiera
Standard qwerty con 66 tasti.
4 tasti di controllo del cursore.
4 tasti di funzione programmati, che possono essere ridefiniti dall'utente per supportare fino a 8 funzioni prescelte. Tasti di controllo del colore.
Tasto di HELP.
Set di caratteri maiuscolo e minuscolo.
Set di caratteri grafici.

Input/Output
User port C16.
Porta seriale Commodore.
Porta per cartuccia Rom e per unità disco parallela.
2 porte per joystick.
Porta di collegamento per unità registratore C1531.
Uscita monitor con segnale composito/croma/luma.
Input/output audio.
Ingresso alimentazione di rete.

Caratteristiche principali
Basic esteso versione 3.5 incorporato, con oltre 75 comandi in grado di pilotare grafica e suono.
Monitor in linguaggio macchina incorporato, con 12 comandi disponibili.
Possibilità di creazione finestre su video.

Periferiche opzionali
SFS 481 unità a disco veloce.
C 1542 unità a disco.
C 1531 registratore a cassette.
MCS 801 stampante a matrice a 7 colori.
MPS 802 stampante a matrice.
DPS 1101 stampante a margherita 'letter quality'.
C 1520 stampante/plotter a 4 colori.
C 1703 monitor a colori.

Altre periferiche
Commodore 16 funziona anche in collegamento con l'unità a disco C 1541, con la stampante a matrice C 1526 e con il monitor a colori C 1701.

Dimensioni
Altezza: 76.2 mm.
Larghezza: 406.4 mm.
Profondità: 203.2 mm.

Alimentazione di rete
240 volts AC, 50 Hz con alimentatore esterno.

Consumo
8.5 watt massimo.

Colore
Grigio antracite.

DOMANDE RISPOSTE

DOMANDE RISPOSTE

Le RAM del Commodore 64

□Quale è l'esatto quantitativo di RAM a disposizione nel mio 64 e perchè con PRINT FRE(0) si ottiene talvolta un valore negativo?

● Sono effettivamente 64K, vale a dire più di 64000 locazioni. Alcune di queste non sono però accessibili all'utente se non lavorando in Linguaggio Macchina. Da un po' di tempo Commodore Computer Club, sta affrontando in modo sistematico questo argomento che consente di sfruttare appieno le potenzialità dei computer Commodore. In questo numero troverai molti articoli su cui documentarti.

Per ciò che riguarda la seconda parte della domanda, tieni presente che se con PRINT FRE(0) si ha un valore positivo, questo rappresenta l'esatto quantitativo disponibile per dati o programmi basic. Se invece il valore è negativo la quantità di RAM disponibile è data dal semplice calcolo:
PRINT 2↑16+FRE(0).

Problemi col comando @0

□Con la mia unità a dischetti 1541, sfruttando molto un floppy (cancellando, sovrapponendo, registrando ecc.), a volte mi accorgo che certi File non sono più riconoscibili e alcuni programmi non appaiono addirittura nella Directory.

● E' molto probabile che hai usato troppo spesso il comando: @0:NOME col quale si «sovrappone» ad un programma già residente su disco, un altro programma con lo stesso nome. Abbiamo notato, infatti, che se questa operazione viene ripetuta troppo spesso succedono pasticci nella B.A.M. che possono rendere, anche se raramente, addirittura inutilizzabile l'intero disco, specie se il programma riscritto con @0 è uno dei primi.

Per evitare malfunzionamenti si consiglia, pertanto, di non utilizzare mai il comando @0, e di procedere alla scrittura di un programma con altro nome e, in seguito, alla cancellazione di quello che non serve più.

Ancora sul 1541

□ A volte, sbagliando a trascrivere qualche programma o file di dati, il LED del drive lampeggia. C'è un modo per spegnerlo senza spegnere e riaccendere il 1541?

● Puoi inizializzare il dischetto:
OPEN 15,8,15,"IØ": CLOSE 15

Oppure ricorrere al comando "U".
Caso del Commodore 64:
OPEN 15,8,15,"U+": CLOSE15
Caso del Vic 20:
OPEN 15,8,15, "U-": CLOSE15

Il tasto Return

□ Dopo aver listato alcune linee basic, mi capita la necessità di corregerne qualcuna e per far questo "salgo" col cursore ed inizio la modifica. A volte, in tale fase, batto però qualche tasto errato. Come posso fare per non "perdere" la vecchia linea?

● La "vecchia" linea rimane lì al suo posto e non le accade nulla fino a che non premi il tasto Return. Se infatti, ad esempio, dopo aver pasticciato la linea, cancelli lo schermo, chiedendo il listato di quella riga ti accorgi che è rimasta immutata, proprio perchè non è stata data la conferma col tasto Return. A volte capita, specie digitando gli apici, (= le virgolette) che il tasto di cancellazione schermo non provoca l'effetto desiderato. E' sufficiente allora premere insieme i tasti Shift e Return. Anche in questo modo la riga rimane immutata.

Accensione di un sistema

□ Iniziando un lavoro al computer, quale apparecchio accendere per primo e quale per ultimo?

● In genere si accende dapprima il computer, poi il drive e per ultima la stampante. Questa però non è una regola generale, tanto è vero che sul manuale di alcuni package di software si consiglia una procedura diversa. Chi scrive, ad esempio, è costretto ad accendere prima il 1541, e poi il Vic. Facendo al contrario, infatti, il drive parte e non si ferma più, chissà perchè. Un altro lettore, a proposito, mi disse che fu costretto a sostituire la lampada al neon della sua camera perchè non appena la accendeva resettava il computer!

In genere, comunque, è bene tenere accese tutte le periferiche che costituiscono il sistema, anche se non devono essere usate. Una cosa da non fare *per nessun motivo*, è quello di collegare i connettori, schede di espansioni, cartucce di Firmware ed altri apparecchi mentre le macchine sono accese. Ho visto saltare più di un computer con questo metodo.

Sui puntatori dei programmi basic

□Mi hanno detto che è possibile "spostare" in alto o in basso, quasi a piacimento, la zona di memoria RAM in cui allocare i programmi basic. Perchè non dite in che modo farlo?

● Ho già affrontato tale argomento sul N.42 (giugno '84) della rivista Micro & Personal Computer nell'articolo "Spostamento dei programmi basic".

Alessandro de Simone

commodore

COMPUTER CLUB

La rivista degli utenti di sistemi Commodore

Presenta

Settembre 1984 - N. 3 - Lire 3.000 Anno I - N. 3 - Sped. Abb. post. gr. III/70 - CR - Distr. Me Pe

COMMODORE

MENSILE PER UTENTI DI VIC 20 & COMMODORE 64

LO SPACCA BYTE

◆

IL C.I.A.

◆

BED

◆

LE RETTE

◆

COMPACTOR

◆

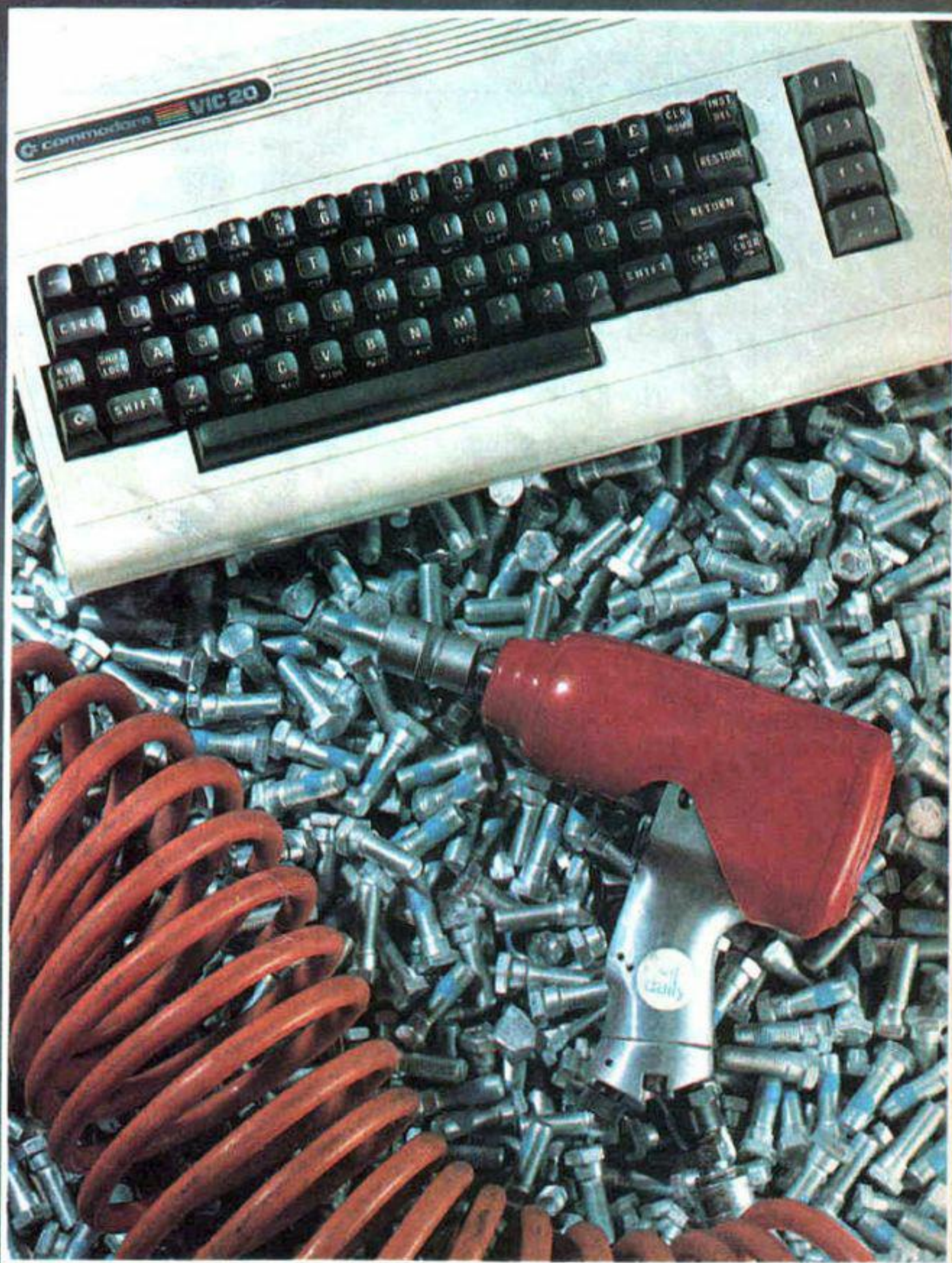
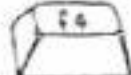
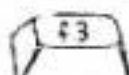
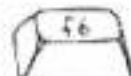
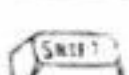
MCD e mcm

◆

SPOSTA BASIC

◆

SUPERLIFE

systems
Communication

Come ottenere i caratteri speciali dei computer Commodore

In molti listati pubblicati in questa rivista figurano alcuni caratteri "speciali". Indichiamo, su questa pagina, il modo di ottenerli.

Simbolo	Codice	Tasti	Simbolo	Codice	Tasti
	= [NERO]	CTRL 1 BLK		= [ARANC]	C= 1 BLK
	= [BIANCO]	CTRL 2 WHT		= [MARR]	C= 2 WHT
	= [ROSSO]	CTRL 3 RED		= [ROSA]	C= 3 RED
	= [AZZUR]	CTRL 4 CYN		= [GRIGIO1]	C= 4 CYN
	= [VIOLA]	CTRL 5 PUR		= [GRIGIO2]	C= 5 PUR
	= [VERDE]	CTRL 6 GRN		= [VERDE]	C= 6 GRN
	= [BLU]	CTRL 7 BLU		= [CELESTE]	C= 7 BLU
	= [GIALLO]	CTRL 8 YEL		= [GRIGIO3]	C= 8 YEL
	= [TF2]	f2		= [TF1]	f1
	= [TF4]	f4		= [TF3]	f3
	= [TF6]	f6		= [TF5]	f5
	= [TF2]	f8		= [TF7]	f7
	= [DOWN]	CRSR		= [UP]	SHIFT CRSR
	= [RVS]	CTRL 9 RVS ON		= [RVOFF]	CTRL 0 RVS OFF
	= [HOME]	CLR HOME		= [CLEAR]	SHIFT CLR HOME
	= [RIGHT]	CRSR		= [LEFT]	SHIFT CRSR
	= [DEL]	INST DEL	π	= [PI]	SHIFT ↑ π

Precisazioni su programmi pubblicati

Fascicolo 11

Programma "Scrittura a tutto schermo"
Riga 270 invece di IF R=147, leggasi IF P=147

Programma "Caleidoscopio"
Il listato non contiene errori di sorta. Molti lettori, però, commettono alcuni errori di battitura.

Nella riga 150 il segno che compare in ASC ("-") non è il segno meno, ma il carattere semigrafico che si ottiene digitando Shift e l'asterisco (*).

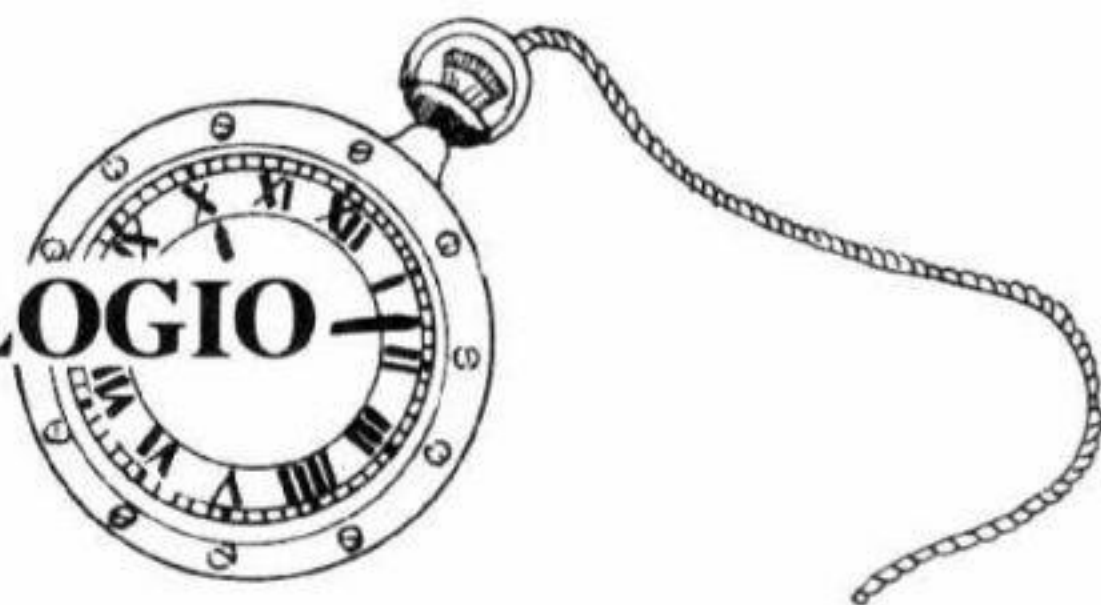
Anche nella riga 170 il carattere che compare non è il segno "+" ma il carattere semigrafico che si ottiene con Shift insieme con "+".

Programma "Easycalc"
Il listato non contiene errori di sorta. Si tenga presente che utilizza la tecnica della manipolazione del buffer di tastiera e pertanto, durante l'elaborazione dei dati, è necessario non battere alcun tasto della tastiera.

Lo stesso programma è stato inserito, col nome PRESTOCALC, nella nuova rivista su cassetta COMMODORE CLUB.

Programma "Archivio dischi"
Il programma gira perfettamente. Eventuali errori di sintassi sono dovuti al fatto che si è fatto ricorso all'abbreviazione "?≠" al posto di PRINT≠. Tale abbreviazione, dovrebbe esser noto, è illecita.

Fascicolo N. 12

Programma "Battaglia navale"
La riga 1190 va così corretta:
1190 IF A$ ="S" THEN RUN

OROLOGIO

Questo programma che utilizza le routines grafiche di Danilo Toma, pubblicate su C.C.C. N.10, trasforma il vostro computer in un orologio analogico.

Dopo aver digitato il programma, salvatelo, caricate le routines di Toma,lanciatele e caricate questo programma. Dopo il RUN il computer vi chiederà se avete già caricato le routines grafiche e, in caso affermativo, dovrete solo premere il tasto RETURN per continuare. Ora il computer vorrà sapere se volete modificare l'ora o se volete continuare con quella già impostata: infatti, il C.64 continua a svolgere la sua funzione di orologio fino allo spegnimento della macchina. Potrete quindi fermare il programma o caricarne altri senza «perdere» l'ora impostata.

Avete anche la possibilità di attivare, o meno, la sveglia. Con essa, su dodici ore, vanno introdotte le ore e i minuti separati da una virgola. Ricordatevi che le ore vanno da 0 a 11 e i minuti da 0 a 59; e, quindi, per il computer, non esistono le 12 che, d'altra parte, vanno indicate come ore 0.

Se volete cambiare l'ora, sullo schermo comparirà un orologio con le lancette sovrapposte e lampeggianti indicanti le 0,0 (mezzogiorno). Premendo il tasto F5 si otterrà l'avanzamento dei minuti, con F6 si avrà un avanzamento ancora dei minuti a passi, in questo caso, di cinque. Con F7 si produrrà uno spostamento della lancetta delle ore. Con F1, l'orologio parte.

Se volete cambiare l'ora quando l'orologio è già attivo premere F3. Premendo F4, vi sarà chiesto se volete la sveglia e, in caso affermativo, a che ora.

Funzionamento

Se avete letto l'articolo di Toma ricorderete che un orologio è quanto di più sconsigliato si possa fare utilizzando quelle routines. Esse, infatti, ogni volta che vengono chiamate, fermano il conteggio dell'orologio software del BASIC contenuto nella variabile TI$. Per ovviare a questo inconveniente è stato necessario ricercare tra i vari integrati un orologio di tipo hardware. E, come sempre, mamma Commodore ha pensato a tutto munendo il C 64 di ben due di questi orologi. Essi sono contenuti nelle due CIA (circuito adattatore di interfaccia) che sono collegate alla tastiera, alla user port e all'interfaccia seriale.

Ognuna di esse contiene un orologio di 24 ore AM/PM con una precisione di 1/10 di secondo organizzato su quattro Bytes: ore, minuti, secondi e decimi di secondo. L'indicatore AM/PM, non usato in questo programma, si trova sul bit più significativo delle ore. I valori contenuti in questi bytes non sono numeri codificati in esadecimali, ma in codice BCD e necessitano quindi di una conversione. L'orologio è 'clockato' direttamente dalla corrente alternata di rete (portata preventivamente a valori TTL) e, dato che il computer è predisposto anche per gli standard americani, deve essere programmato per funzionare a 50 Hz.

Per una corretta programmazione o lettura dell'orologio è necessario rispettare una procedura ben precisa. Ogni volta che viene effettuata una operazione di lettura nel registro delle ore, tutti i dati vengono congelati fino a quando non viene effettuata la lettura dei decimi di secondo. Questo serve a far sì che non si verifichino errori di lettura causati da operazioni di riporto. Nel caso debba essere letto un solo registro (come nel caso dei secondi) non potendosi verificare un errore di riporto, è possibile fare una lettura 'al volo' dei dati. Per la scrittura delle ore la sequenza da rispettare è molto simile: ogni volta che viene 'pokeato' un valore nel registro delle ore dell'orologio si ferma e riparte solo dopo la programmazione dei decimi di secondo. Questo assicura che l'orologio parta sempre al momento desiderato.

Per i più appassionati di PEEK & POKE posso segnalare l'esistenza di un allarme abbinato a ciascun orologio. Quest'ultimo può essere selezionato settando il bit più significativo del registro di controllo B della porta. Una volta settato sarà possibile programmare l'allarme nello stesso modo e nelle stesse locazioni dell'orologio. Per riprogrammare l'orologio sarà quindi necessario riportare alla condizioni logica 0 il bit in questione. Tale allarme è, come potrete capire, difficilmente utilizzabile dal BASIC perchè aziona la sveglia generando un'interruzione.

La sveglia usata in questo programma, invece, è ottenuta con una semplice linea if... then.

Andrea & Alberto Boriani

(segue listato)

Leoni informatica

HARDWARE & SOFTWARE HOUSE

HOTLINE

linea telefonica dedicata alla risoluzione dei problemi dei clienti. Chiamando il numero telefonico riservato che troverete sulla cartolina garanzia acclusa ai programmi, riceverete tutte le informazioni che vi necessitano.

UPDATE

servizio di aggiornamento continuo dei programmi acquistati. Ogni modifica ai programmi realizzati dalla Leoni Informatica sarà fornita agli utenti degli stessi.

GARANZIA

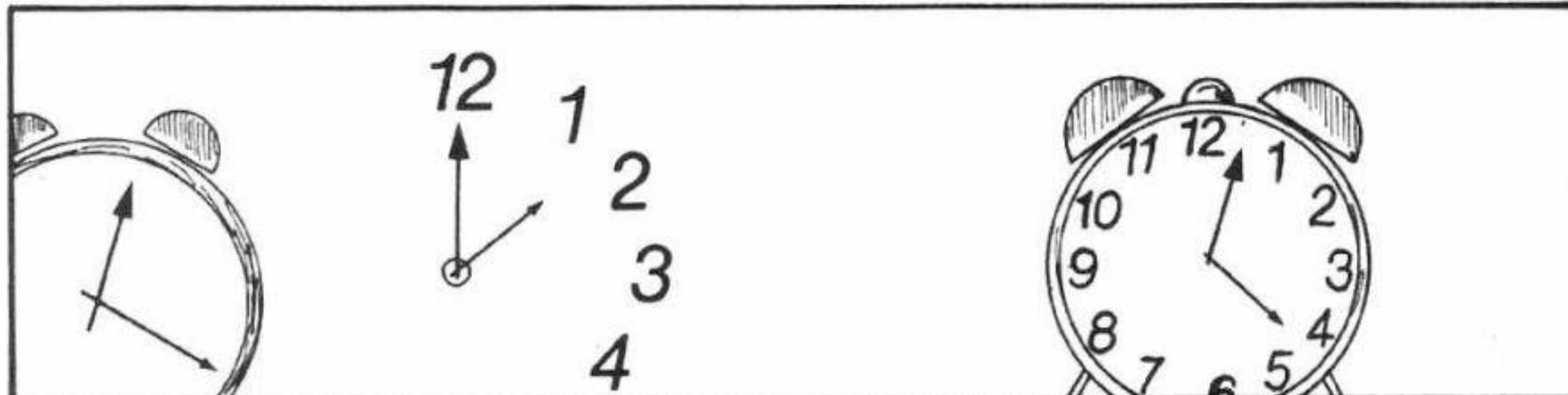
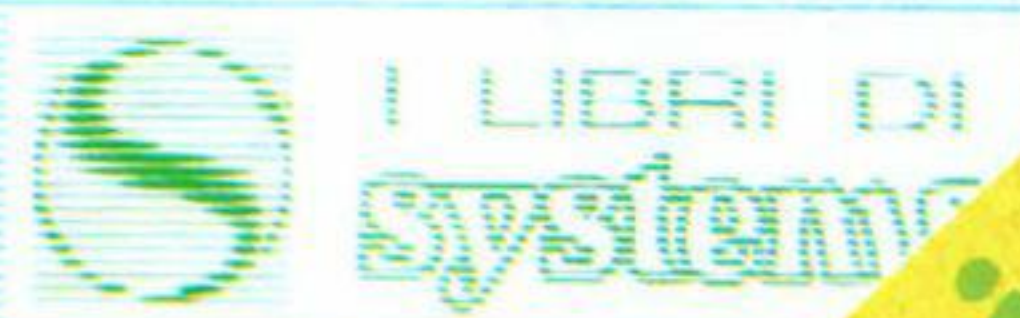
tutti i programmi Leoni Informatica sono coperti da garanzia a Vita contro guasti di origine.

COMMODORE 64 SOFTWARE

Programmi in configurazione base (*) IVA esclusa

cod.	DESCRIZIONE	Prezzo	cod.	DESCRIZIONE	Prezzo
		PERSONALI			
0046	Ammortamento Mutui	60.000	0049/T	Totoplus	100.000
0050/T	Totocalcio a sviluppo colonnare	80.000	0051/T	Gestione dei Conti di casa	100.000
0058/T	Calcolo dell'equo canone	60.000	0055/T	Impariamo il BASIC	100.000
0060/T	Modello 740 quadro N	80.000	0059/T	Modello 740 Quadro O	80.000
0066	Conto corrente	150.000	0063	Cento programmi BASIC	80.000
0176	Diary 64 (Commodore)	100.000	0091/T	Rubrica telefonica	100.000
0174	Corso di Dattilografia	100.000			
		GESTIONI GENERALI			
0047	Anagrafiche	250.000	0056	Dichiarazione I.V.A.	60.000
0065/T	Fido Clienti	100.000	0067	Piano dei Conti (cli/for/gen)	150.000
0068	Appuntamenti	150.000	0071	Ordini (cli/for)	150.000
0090	Mailing List (riordino alfabetico. cap. prov.)	150.000	0094	Scheda 4800 car. (cli/for/rapp./az. etc.)	160.000
0096	Scheda 4800 car. agganciata al Mailing	250.000	0097	Super Mail (5 chiavi accesso, riordini..)	250.000
0116	Scadenziario effetti (ric. bancarie, tratte, etc.)	200.000	0120	Contabilita' fatture (Iva, impon. etc.)	250.000
0121	Contabilita' Semplice (Tratte/Fatt/Conti/etc.)	400.000	0160	Bolle e Fatture	200.000
0125	Contabilita' Generale (132 colonne)	annunciato			
		GESTIONI SPECIFICHE			
0045	Agenti e Rappresentanti	250.000	0048	Scadenziario premi e polizze	150.000
0164	Agenzie Immobiliari	250.000	0086	Librerie e biblioteche	120.000
0148	Studi Ottici	300.000	0151	Farmacie	300.000
0149	Studi Dentistici	300.000	0152	Studi Medici	300.000
0131	Hotel e Pensioni	400.000	0132	Parrucchieri	400.000
0133	Gommisti	400.000	0134	Clubs Nautici	250.000
0135	Officine	400.000	0171	Ristoranti	400.000
0170	Tavola Calda	400.000	0172	Lavanderie	400.000
0175	Condominio New	annunciato			
		GESTIONE TESTI			
0190	Hes Writer	70.000	0191	Word Processor III	100.000
0192	Bank Street Writer	70.000	0319	Easy Script (Commodore)	75.000
		TECNICI			
0136	Legge 373 (calcolo degli isolamenti termici)	150.000	0140	Ingegneria civile I (calcoli strutt.)	250.000
0141	Ingegneria civile II (travi intelaiate)	250.000	0404	Computo metrico	300.000
0409	Diagnostica C64	40.000			
		MAGAZZINI			
0142	Magazzino e Fatturazione semplici	200.000	0143	Magazzino Grossisti (2500 art.)	380.000
0144	Magazzino e Fatturazione agganciati	400.000	0158	Magazzino Dettaglio (2500 art.)	380.000
0148	Magazzino alfa dett. (600 art.)	250.000	0159	Magazzino Taglia/Col. (2500 art.)	380.000
		LINGUAGGI & UTILITIES			
0162/T	Screen Grafix (Abacus)	85.000	0163	Copia Disco singolo	50.000
0064	Petspeed Compiler (Commodore)	80.000	0165/T	Zoom	70.000
0167	Simon's Basic (Commodore)	85.000	0168/T	Turbo Tape	50.000
0177	Pilot (linguaggio)	70.000	0178	Ultra Basic	125.000
0179	Character Editor	70.000	0193/C	Basic 4.0	80.000
0194	Sprite Generator	70.000	0195	Assembler	80.000
0196	S.A.M. (Tronix)	150.000	0197	G-Pascal	95.000
0198	Forth 64 (Commodore)	95.000	0199	Tool 64 (Commodore)	85.000
0200	Master 64 (Commodore)	145.000	0201/C	Scheda CP/M (Commodore)	125.000
		GESTIONE DATI			
0157	Calc Result (Handic)	195.000	0161	Data Base Manager (Italiano)	120.000
0205	Super Base (Commodore)	175.000	0206	Magic Desk (Commodore)	75.000
0207	Koala Joystick	130.000	0209	Data Log.	020.000

Leoni informatica

Direzione Commerciale - via Don Rodrigo, 6 - 20142 MILANO tel. 02/8467378
Direzione assistenza tecnica: 20143 MILANO - via Valsolda, 21
Negozio vendite al pubblico: Milano - Corso di Porta Romana, 123

```
10 REM*********************************
11 REM*                               *
20 REM*   C O M M O D O R E    6 4    *
21 REM*                               *
22 REM* OROLOGIO ANALOGICO CON SVEGLIA*
23 REM*                               *
30 REM* ALBERTO       &       ANDREA  *
40 REM*                               *
50 REM*           BORIANI             *
60 REM*                               *
70 REM*          02/6195626           *
80 REM*                               *
90 REM*********************************
95 :
100 X%=0:Y%=0:X1%=0:Y1%=0
110 SC=49763:DR=49728:PL=49693:CL=49785:MO=49821
120 INPUT"█HAI CARICATO LE ROUTINES GRAFICHE  SI████";A$
130 IFLEFT$(A$,1)="N"THENPRINT"CARICALE":STOP
140 IFLEFT$(A$,1)<>"S"THEN120
150 RA=.0174532925:POKE56590,128
160 INPUT"████████████VUOI CAMBIARE L'ORA  SI████";P$
170 IFLEFT$(P$,1)="N"THEN200
180 IFLEFT$(P$,1)<>"S"THEN160
190 POKE56587,0:POKE56586,0:POKE56584,0
200 INPUT"████████████VUOI LA SVEGLIA  SI████";O$
210 IFLEFT$(O$,1)="N"THEN240
220 IFLEFT$(O$,1)<>"S"THEN200
230 INPUT"████████████SVEGLIA";Y,U
240 SYS SC
250 POKE251,1:POKE252,6:SYSCL
260 POKE248,1:SYSMO:POKE2,1
270 FORH=0TO2*πSTEPπ/6
280 X%=159+SIN(H)*70:Y%=99+COS(H)*60
290 X1%=159+SIN(H)*83:Y1%=99+COS(H)*73:SYSDR
300 NEXTH
310 :
320 FORH=0TO2*πSTEPπ/30
330 X%=159+SIN(H)*70:Y%=99+COS(H)*60
340 X1%=159+SIN(H)*74:Y1%=99+COS(H)*64:SYSDR
350 NEXTH
```

```
360 REM**********************************
370 REM******** MAIN *****************
380 REM**********************************
390 IFLEFT$(P$,1)="S"THEN860
400 GOSUB540:GOSUB690
410 AS=PEEK(56585)
420 IFAS=DFTHEN410
430 POKE2,0:SYSDR:GETG$:IFG$="■"THEN860
440 AD=PEEK(56586)
445 IFG$="■"THENP$="N":POKE248,0:SYSMO:GOTO200
450 IFY=FJ ANDU=FKANDASC(O$)=83THENGOSUB1010
460 IFFH=ADTHEN400
470 X%=159:Y%=99:POKE2,0
480 X1%=JX%:Y1%=JY%:SYSDR:X1%=OX%:Y1%=OY%:SYSDR
490 GOSUB540
500 GOTO400
510 REM**********************************
520 REM** PLOTTAGGIO MINUTI ***********
530 REM**********************************
540 X%=159:Y%=99:POKE2,1
550 FG=PEEK(56587):FH=PEEK(56586):Z=PEEK(56584)
560 FJ=(FGAND15)+.625*(FGAND240):REM ORE
570 FK=(FHAND15)+.625*(FHAND240)
580 Y1%=99+COS(FK*RA*6)*48
590 X1%=159+SIN(FK*RA*6)*58:SYSDR
600 JX%=X1%:JY%=Y1%
610 REM**********************************
620 REM***PLOTTAGGIO ORE *************
630 REM**********************************
640 Y1%=99+COS((FJ*60+FK)*RA/2)*35
650 X1%=159+SIN((FJ*60+FK)*RA/2)*45:SYSDR
660 OX%=X1%:OY%=Y1%
670 RETURN
680 REM**********************************
690 REM*** PLOTTAGGIO SECONDI *********
700 REM**********************************
710 DF=PEEK(56585):POKE2,1
720 TR=(DFAND15)+.625*(DFAND240)
730 Y%=99+COS(TR*RA*6)*54
740 X%=159+SIN(TR*RA*6)*64
750 :
760 Y1%=99+COS(TR*RA*6)*57
770 X1%=159+SIN(TR*RA*6)*67:SYSDR
780 RETURN
790 REM**********************************
800 REM* QUESTO PROGRAMMA UTILIZZA LE *
810 REM* ROUTINE GRAFICHE DI D. TOMA  *
820 REM*     PUBBLICATE NEL N.10 DI    *
830 REM* COMMODORE   COMPUTER   CLUB  *
840 REM**********************************
850 :
```

```
860 GOSUB540:REM*********************************
870 REM*******************************
880 GETA$
890 IFA$="■"THENDI=DI+1
900 IFA$="■"THENDO=DO+1
910 IFA$="■"THENDI=DI+5
920 IFDI>=60THENDI=0:DO=DO+1
930 DO=-(DO<=11)*DO
935 IFA$="■"THENPOKE56585,0:POKE56584,2:GOTO490:STOP
940 HO=(DO-INT(DO/10)*10)+16*INT(DO/10)
950 MI=(DI-INT(DI/10)*10)+16*INT(DI/10)
960 POKE56587,HO:POKE56586,MI
970 X%=159:Y%=99:POKE2,0
980 X1%=JX%:Y1%=JY%:SYSDR:X1%=OX%:Y1%=OY%:SYSDR
990 GOSUB540
1000 GOTO880
1010 IFO=15THENRETURN
1020 O=O+1:POKE54276,0:POKE54277,0:POKE54272,0
1030 POKE54296,15
1040 POKE54277,90
1050 POKE54276,17
1060 POKE54273,57:POKE54272,37
1070 RETURN

READY.
```

IL SISTEMA "MUSIC 64"

Che il Commodore 64 abbia notevoli capacità musicali è un fatto più che noto. Altrettanto nota, purtroppo, è la notizia che la programmazione del circuito integrato che provvede alla riproduzione dei suoni è decisamente ardua e molti sono i lettori che, dopo alcuni tentativi, rinunciano alla... produzione artistico-musicale.

La ditta Novel International di Recanati sembra aver trovato la soluzione al problema degli audiofili che posseggono un Commodore 64 e desiderano cimentarsi nell'eseguire brani musicali.

Non è il solito programma che consente, premendo i tasti del personal computer, di riprodurre note musicali. E' invece un completo sistema Hard/Soft che permette di utilizzare una autentica tastiera che, tramite un adattatore da inserire sul retro del 64, utilizza le complesse potenzialità del computer stesso. I 49 tasti, che comprendono 4 ottave da DO a DO, sono infatti a passo professionale e non presentano problemi per chi sa suonare su di un organo elettronico.

Il bello, però, come immaginerete, non

è solo nell'hardware ma nel software che permette di utilizzare la tastiera in infiniti modi. I programmi presenti nella confezione, infatti, non sono «protetti» e possono essere modificati a piacimento dall'utente. Se però non si è in grado di intervenire sui listati, niente paura: la tastiera può essere utilizzata senza sapere nulla di programmazione o di computer.

Dei due programmi disponibili, il primo permette l'uso della tastiera in modo polifonico, vale a dire che premendo più tasti vengono fuori più note musicali alla volta in modo da formare accordi e suonate a due mani, proprio come nei «normali» organi. Caricato il primo programma è possibile scegliere fra cinque timbri: Spinet, Accordion, Tabular bell, Flute, Banjo. Ricordiamo che il menu può essere ampliato a piacere dall'utente a patto, ovviamente, che sia un programmatore. Il bello, però, consiste nel fatto che di ogni timbro selezionato è possibile modificare alcuni (o tutti) i parametri premendo semplicemente dei tasti. In altre parole se selezionate FLUTE è possibile trasformare il suono rendendolo simile ad un oboe, ad un pianoforte o ad altri strumenti... immaginari intervenendo sull'Attack o sul Decay o sul Release e, in pratica, su tutti i parametri su cui poter intervenire. In effetti le cinque opzioni che possono esser selezionate sono state messe solo per venire incontro all'utente. Se questi non è pienamente soddisfatto del suono riprodotto può modificarlo come gli pare. I parametri su cui si può agire sono otto: Attack, Decay, Sustain, Release, Volume, Transpose, Waveform, Duty cicle. Se si combina qualche pasticcio è sufficiente premere un tasto perchè le condizioni iniziali vengano ripristinate. Se si realizza un timbro interessante sarà sufficiente trascrivere i valori visualizzati e riportarli nuovamente quando si utilizza il computer una seconda volta.

Col secondo programma, che consente l'utilizzo della tastiera in modo monofonico, le possibilità di intervenire sul timbro sono maggiori, dato che è possibile alterare i valori relativi ad altri parametri oltre quelli già visti. Senza dilungarci oltre ricordiamo che in questo caso i timbri base sono ben tredici ed anche in questo caso è possibile alterare a piacimento i vari parametri ed ottenere riproduzioni sonore che hanno, molto spesso, dell'incredibile. Tra gli altri basterà indicare l'effetto wha-wha, quello chitarra e uno particolare denominato Random che sembra... suonare da solo.

In conclusione il modello della Novel può rappresentare una interessante alternativa per quei gruppi musicali che, disponendo già di un Commodore 64, vogliano studiare nuovi «sound» da utilizzare nei propri brani. Se poi nel gruppo c'è qualcuno in grado di sviluppare programmi particolari, sarà una piacevole sorpresa scoprire che non solo i gruppi rock più affermati possono disporre di versatili sintetizzatori musicali.

Alessandro de Simone

DAMA CINESE

Un solitario che si gioca su una scacchiera a forma di croce.

Bisogna mangiare tutte le pedine: se il gioco riesce, ne deve rimanere solo una.

Si può mangiare solo in orizzontale e verticale: il programma, comunque, non accetta le mosse sbagliate.

Per muovere, bisogna indicare la casella di partenza e quella d'arrivo. Va indicata prima la lettera e poi il numero per le due caselle desiderate.

Esempio: ? F4D4 (Return).

La mossa può essere corretta col tasto 'Inst/Del'. Sul video, per ogni mossa, compare il numero di pedine rimaste ed il tempo trascorso dall'inizio della partita.

Giancarlo Castagna

➡

```
100 REM ******************************
110 REM * DAMA CINESE:  SOLITARIO    *
120 REM *       PER COMMODORE 64     *
130 REM *     GIANCARLO CASTAGNA     *
140 REM *     VIA  S.GIMIGNANO 9     *
150 REM *        20146 MILANO        *
160 REM *      TEL.  02/4155768      *
170 REM ******************************
180 PRINT"♥"
190 POKE53281,6  :POKE53280,0
200 PRINT"▓▓▓▓▓▓▓▓▓▓"
210 S1$="▓▓▓ ▓▓▓ ▓   ▓ ▓▓▓ ▓▓▓ ▓▓▓ ▓ ▓▓▓"
220 S2$="▓   ▓ ▓ ▓   ▓ ▓   ▓ ▓ ▓ ▓ ▓ ▓"
230 S3$="▓▓▓ ▓ ▓ ▓   ▓ ▓   ▓▓▓ ▓▓▓ ▓ ▓ ▓"
240 S4$="  ▓ ▓ ▓ ▓   ▓ ▓   ▓ ▓ ▓ ▓ ▓ ▓ ▓"
250 S5$="▓▓▓ ▓▓▓ ▓▓▓ ▓ ▓   ▓ ▓ ▓ ▓ ▓ ▓▓▓"
260 PRINT"▓▓▓▓"S1$:PRINT"▓▓▓▓"S2$:PRINT"▓▓▓▓"S3$
270 PRINT"▓▓▓▓"S4$:PRINT"▓▓▓▓"S5$
280 PRINT"▓▓▓▓▓▓▓▓▓▓PREMI IL TASTO 'F7'"
290 GETF$:IFF$=""THEN290
300 IFF$<>"▓"THEN280
310 IFF$="▓"THENPRINT"♥"
320 POKE54296,15:POKE54278,248:POKE54276,17
330 SI=54273:W=80:W1=60
340 DIMM(7,7):DIMN(9,9)
350 FORR=1TO7:FORC=1TO7
360 M(R,C)=1:NEXT:NEXT
370 M(4,4)=0
380 FORK=1TO9:FORH=1TO9
390 N(K,H)=1:NEXT:NEXT
400 N(5,5)=0
410 FORC=1TO9:N(1,C)=0:NEXT
420 FORC=1TO9:N(9,C)=0:NEXT
430 FORK=1TO9:N(K,1)=0:NEXT
440 FORK=1TO9:N(K,9)=0:NEXT
450 N(2,2)=0:N(2,3)=0:N(3,2)=0:N(3,3)=0
460 N(7,7)=0:N(7,8)=0:N(8,7)=0:N(8,8)=0
470 N(2,7)=0:N(2,8)=0:N(3,7)=0:N(3,8)=0
480 N(7,2)=0:N(7,3)=0:N(8,2)=0:N(8,3)=0
490 A1$="┌─┬─┬─┐"
500 A2$="│ │ │ │"
510 A3$="├─┼─┼─┤"
520 A4$="┌─┬─┼─┼─┼─┬─┐"
530 A5$="│ │ │ │ │ │ │ │"
540 A6$="├─┼─┼─┼─┼─┼─┼─┤"
550 A7$="└─┴─┼─┼─┼─┴─┘"
560 A8$="└─┴─┴─┘"
570 B$="ABCDEFG"
580 B1$="1 2 3 4 5 6 7"
590 PRINT"▓▓▓▓▓"
600 PRINTSPC(10)A1$:PRINTSPC(10)A2$
610 PRINTSPC(10)A3$:PRINTSPC(10)A2$
620 PRINT"▓▓▓▓▓▓"A4$:PRINT"▓▓▓▓▓▓"A5$
630 PRINT"▓▓▓▓▓▓"A6$:PRINT"▓▓▓▓▓▓"A5$
640 PRINT"▓▓▓▓▓▓"A6$:PRINT"▓▓▓▓▓▓"A5$:PRINT"▓▓▓▓▓▓"A7$
650 PRINTSPC(10)A2$:PRINTSPC(10)A3$
660 PRINTSPC(10)A2$:PRINTSPC(10)A8$
670 PRINT"▓▓▓"
680 PRINTSPC(7)B1$
690 PRINT"▓▓▓▓▓"
700 FORI=1TO7:PRINTSPC(4)MID$(B$,I,1):PRINT:NEXT
710 PRINTSPC(7)B1$
```

```
720 C$="■ ■"
730 PRINT"■■■■■■"
740 FORJ=1TO7:PRINTSPC(11)C$;:PRINTSPC(1)C$;
750 PRINTSPC(1)C$:PRINT:NEXT
760 PRINT"■■■■■■■■■■■"
770 FORI=1TO3:PRINTSPC(7)C$;:PRINTSPC(1)C$;
780 PRINTSPC(7)C$;:PRINTSPC(1)C$:PRINT
790 NEXT
800 D$="  "
810 PRINT"■■■■■■■■■■■■■■"
820 PRINTSPC(13)D$
830 PRINT"■■■"
840 PRINTSPC(27)"■MOSSA■"
850 PRINT"■■■";:PRINTSPC(27)"■PEDINE■"
860 PRINTSPC(27)"■RIMASTE■"
870 PR=32
880 PRINT"■■■■■■■■■■■■■■";
890 PRINTSPC(27)"■ "PR"■ ■"
900 PRINT"■■";:PRINTSPC(27)"■TIME■"
910 TI$="000000"
920 GOSUB1130:PRINT"■"
930 FORK=4TOI+4:PRINT"■":NEXT
940 PRINTSPC(5);:PRINTSPC(J*2) D$
950 PRINT"■"
960 FORKK=4TOII+4:PRINT"■":NEXT
970 PRINTSPC(5);:PRINTSPC(JJ*2)C$
980 PRINT"■"
990 FORZ=4TO(I+II)/2+4:PRINT"■":NEXT
1000 PRINTSPC(5);:PRINTSPC((J+JJ)/2*2)D$
1010 POKESI,W:FORT=1TO50:NEXT:POKESI,0
1020 POKESI,W1:FORT=1TO100:NEXT:POKESI,0
1030 PR=PR-1
1040 PRINT"■■■■■■■■■■■■■■";
1050 PRINTSPC(27)"■ "PR"■ ■"
1060 TT$=TI$
1070 T1$=MID$(TT$,3,2):T2$=RIGHT$(TT$,2)
1080 PRINT"■■■■■";:PRINTSPC(27)"■"T1$"."T2$"■"
1090 IFPR<15THENGOSUB1420
1100 IFX=1ANDPR=1THEN1570
1110 IFX=PRTHEN1500
1120 GOTO920
1130 REM *** INPUT SOBROUTINE       ***
1140 REM ****************************
1150 FORI=1TO8:POKE1250+I,32:NEXT
1160 PRINT"■■■■■■"
1170 PRINTSPC(27);:INPUTI$
1180 L$="ABCDEFG"
1190 FORK=1TO7
1200 IFMID$(I$,1,1)=MID$(L$,K,1)THENI=K
1210 NEXTK
1220 J$=MID$(I$,2,1):J=VAL(J$)
```

CERCASI

La redazione di Commodore Computer Club vuole potenziarsi e ricerca collaboratori part-time e insegnanti di discipline scientifiche e tecniche preferibilmente residenti nell'area di Milano

Ai collaboratori che stiamo ricercando verrà richiesto di collaborare alle varie iniziative della casa editrice con articoli, libri, raccolte di programmi e l'italianizzazione di software, di cui abbiamo i diritti d'autore, orientati alla didattica per le scuole medie e superiori.
I prescelti, pertanto dovranno possedere un sistema completo di Vic 20 oppure Commodore 64 e sapere programmare sia in basic che in linguaggio macchina. La conoscenza dell'inglese e di altri personal computer è un titolo preferenziale.

Compensi

Tutti i lavori svolti su incarico della redazione verranno sempre compensati in base ai miglior standard di mercato.

Primo contatto

Per incontrarci telefonate allo 02/8467348 chiedendo della signorina Piera

```
1230 FORKK=1TO7
1240 IFMID$(I$,3,1)=MID$(L$,KK,1)THENII=KK
1250 NEXTKK
1260 JJ$=MID$(I$,4,1):JJ=VAL(JJ$)
1270 IFM(II,JJ)=1THENGOSUB1690:GOTO1130
1280 IFM((I+II)/2,(J+JJ)/2)=0THENGOSUB1690:GOTO1130
1290 IFII=I+2ANDJJ=J+2THENGOSUB1690:GOTO1130
1300 IFII=I-2ANDJJ=J-2THENGOSUB1690:GOTO1130
1310 IFII=I+2ANDJJ=J-2THENGOSUB1690:GOTO1130
1320 IFII=I-2ANDJJ=J+2THENGOSUB1690:GOTO1130
1330 IFII=I+2ORII=I-2THEN1360
1340 IFJJ=J+2ORJJ=J-2THEN1360
1350 GOSUB1690:GOTO1130
1360 GOSUB1390
1370 GOSUB1400
1380 RETURN
1390 M(I,J)=0:M(II,JJ)=1:M((I+II)/2,(J+JJ)/2)=0:RETURN
1400 N(I+1,J+1)=0:N(II+1,JJ+1)=1
1410 N((I+II+2)/2,(J+JJ+2)/2)=0:RETURN
1420 X=0
1430 FORI1=2TO8:FORJ1=2TO8
1440 IFN(I1,J1)=1THEN1460
1450 GOTO1480
1460 IFN(I1+1,J1)=0ANDN(I1-1,J1)=0ANDN(I1,J1+1)=0ANDN(I1,J1-1)=0THENX=X+1
1470 IFX=PRTHEN1500
1480 NEXTJ1:NEXTI1
1490 RETURN
1500 L=L+1:PRINT"█████████████████████████";
1510 PRINT"█████ PUOI ANCORA MIGLIORARE ! █"
1520 IFL=6THENL=0:GOTO1650
1530 FORI=1TO200:POKESI,50:NEXT:POKESI,0
1540 PRINT"□";
1550 PRINT"                                 ":REM 33 SPAZI
1560 FORI=1TO200:NEXT:GOTO1500
1570 L=L+1
1580 PRINT"█████████████████████████";
1590 PRINT"█████ COMPLIMENTI : SEI UN CAMPIONE █"
1600 IFL=6THENL=0:GOTO1650
1610 FORI=1TO200:POKESI,50:NEXT:POKESI,0
1620 PRINT"□";
1630 PRINT"                                 ":REM 33 SPAZI
1640 FORI=1TO200:NEXT:GOTO1570
1650 PRINT"█████ VUOI GIOCARE ANCORA (S/N) ? "
1660 GETQ$:IFQ$=""THEN1660
1670 IFQ$="S"THEN RUN
1680 IFQ$="N"THEN END
1690 POKE54296,15:POKE54278,248:POKE54276,17
1700 SI=54273:W=80:W1=60
1710 POKESI,10:FORI=1TO50:NEXT:POKESI,0
1720 POKESI,20:FORI=1TO200:NEXT:POKESI,0
1730 RETURN
READY.
```

CACCIA AL RAGNO

Questo programma, come dice il titolo, vi permetterà di cimentarvi in una appassionata quanto pericolosa caccia al ragno.

Vediamo un po' le regole del gioco: sulla sinistra dello schermo apparirà la canna del vostro fucile, ad una altezza variabile scelta casualmente. Dall'alto scenderà un ragno, calandosi con la ragnatela. Posizione sullo schermo e velocità di discesa sono anch'esse scelte casualmente.

Per sparare bisogna premere il tasto «space», tenendo presente che il ragno si arresta al momento dello sparo. Se si colpisce il ragno, tutto bene; altrimenti cominciano i guai! Infatti questi sono ragni di una razza particolare, molto intelligente. Se non li colpite, questi furbacchioni vanno a costruire un muro, piazzando un mattone alla volta, proprio davanti al vostro fucile. Più il muro diventa alto, più difficile diventerà per voi poter sparare.

Potrà capitare, infatti, che il fucile venga posizionato ad una altezza inferiore a quella del muro. In questo caso il ragno potrà scendere indisturbato, e sistemare un altro mattone sul muro, aumentando ulteriormente il vostro handicap.

Anche per voi però c'è una possibilità: i vostri proiettili. di una lega speciale, sono in grado di sgretolare un mattone in due colpi. Il primo colpo comprime il mattone, mentre il secondo lo attraversa come se niente fosse. Se vi trovate di fronte il muro, quindi, sparate lo stesso! Se in futuro vi troverete nella stessa posizione, potrete sparare al ragno forando il muro. Ma attenzione: il prossimo ragno che non colpirete, andrà a tappare il buco, invece di piazzare il mattone sulla sommità del muro.

I ragni in totale sono 20, ed a turno scenderanno per costruire il muro. Se li uccidete tutti, avete vinto; se invece il muro raggiunge una certa altezza, tale da non permettervi più di sparare, allora i ragni prendono il sopravvento e voi avete perso. In alto a destra apparirà il numero dei ragni uccisi, a sinistra il numero dei colpi sparati. Se volete interrompere il gioco, ricordatevi di premere il tasto «Q».

```
100 REM CACCIA AL RAGNO
110 REM PER VIC 20 INESPANSO
120 :
130 S1=36874:S2=S1+1:S3=S1+2:S4=S1+3
140 PU=4:V=5:POKE36879,30:POKE36878,15
150 R$="[illegible]"
160 C$="[illegible]"
170 D=22:TR=20:W=30720:GOSUB180:GOTO210
180 PRINT"[illegible]";:X$="[illegible]":FORI=1TOD:PRINTX$;:NEXT
```

Con i nuovi programmi itali
Commodore 64 ser

Per tutti i possessori, anche futuri, del personal computer Commodore 64 c'è un'importante novità: la creazione di una vasta serie di nuovi programmi realizzati espressamente per le esigenze italiane.

Con questi programmi i tecnici italiani della Sirius Elettronica mettono a disposizione un servizio che trova Utenti nei più di-

PROGRAMMI SU DISCO PER CBM-64	Prezzo al pubblico (IVA esclusa)
GESTIONE AGENTI RAPPRESENTANTI	29.000
GESTIONE AMMORTAMENTI MUTUI	29.000
GESTIONE ANAGRAFICHE	39.000
GESTIONE SCADENZE POLIZZE	75.000
TOTOCALCIO	23.000
GESTIONE BILANCIO FAMILIARE	23.000
GESTIONE CONDOMINIO	189.000
IMPARIAMO IL BASIC	39.000
GESTIONE STAMPA FATTURE	34.000
CENTO PROGRAMMI PER CBM 64 (didattico)	49.000
GESTIONE FIDO CLIENTI	29.000
GESTIONE CONTO CORRENTE	23.000
GESTIONE CONTI C.F.G.	29.000
GESTIONE APPUNTAMENTI	23.000
GESTIONE ORDINI	75.000
GESTIONE PORTAFOGLIO CLI/FOR	49.000

E lo trovi

PIEMONTE

ALBERTIN s.n.c.
Via Stradella, 60
10147 TORINO

SOFTEC COMPUTER S.r.l.
C.so S. Maurizio, 79
10124 TORINO

TEKNOCOMPUTERS
Via Madama Cristina, 31/C
10125 TORINO

HOBBY SOUND
Corso Traiano, 28/11
10135 TORINO

ROSSI MARCO E FIGLI
C.so Nizza, 16
12100 CUNEO

LIGURIA

MARISA SELANO
Via Carducci, 138
16121 GENOVA

RAPPR.EL di A. TERSO
Via Borgo Ratti, 23/1/R
16132 GENOVA

VIDEON di BOREA
Via Armenia, 15 Cancello
16129 GENOVA

MASSA AGOSTINO s.r.l.
Via Camozzini, 17/R
16158 GENOVA VOLTRI

DIGIT CENTER s.n.c.
Via Assereto, 78
16036 RECCO

S.A.S.A. COM. S.p.A.
Via Argine Sinistro, 82
18100 IMPERIA

ELTRON s.n.c.
Via Lunigiana, 602
19100 LA SPEZIA

LOMBARDIA

AL RISPARMIO di CASTOLDI
V.le Monza, 204
20126 MILANO

A.R.T.E.
P.le Cadorna, 13
20123 MILANO

F.LLI BONAZZI, S.r.l.
Via P. Sarpi, 11
20154 MILANO

BUSCEMI S.d.f.
C.so Magenta, 27
20123 MILANO

BROLETTO GAS di M. LUOSI
Via Broletto, 43
20121 MILANO

C.G.D.
Consorzio Grossisti Dettaglianti
P.za De Angeli, 3
20146 MILANO

GIGLIONI LAURA
Via d'Ovidio, 8
20131 MILANO

GIGLIONI S.r.l.
Via Sturzo, 45
20131 MILANO

MARCUCCI S.p.A.
Via F.lli Bronzetti, 37
20129 MILANO

WORK SYSTEM di M. AMORE
Via Torino, 18
20052 CASSANO D'ADDA (MI)

PENATI s.n.c.
Via Verdi, 28/30
20011 CORBETTA (MI)

MARIANI E COLAGROSSI
V.le Brianza, 23
20036 MEDA (MI)

M.B.M. INFORMATICA SYSTEMS
C.so Roma, 112
20075 LODI (MI)

SENNA GIANFRANCO & C. s.n.c.
Via Calchi, 5
27100 PAVIA

C.A.R.T. di TORRI & PROSDOCIMI
Via Napoleona, 6/8
22100 COMO

FERRERO SERGIO
Via Mameli, 22
21052 BUSTO ARSIZIO (VA)

L'ELETTRONICA
di CANALI GIORGIO s.n.c.
Via Speroni, 14
22100 VARESE

COMM. PERGOLETTO
Via IV Novembre, 56/58
26013 CREMA

PRISMA S.n.c.
Via Chisleri, 55
26100 CREMONA

TECNICA ELIOGRAFICA
CONTABILE
Gall. Gorni
CASALMAGGIORE (CR)

HI-FI TEMPORIN
C.so Genova, 112
27029 VIGEVANO (MI)

VENETO

AREM s.a.s. di POLI e C.
C.so Cavour, 35
37100 VERONA

CEM-DUE s.a.s.
di F. BIDOLI & C.
Via Locatelli, 19
37100 VERONA

FABBRI RAFFAELLO
Via Stella, 17
37121 VERONA

NOVA SYSTEM s.r.l.
Via Disciplina, 9
37036 S. MARTINO B.A. (VR)

DAICOM s.r.l. DAI ZOVI LINO
Via Napoli, 5
36100 VICENZA

ZUCCATO s.r.l.
C.so Palladio, 78
36100 VICENZA

MARANGONI GIUSEPPE e C.
Via Marconi, 8
36015 SCHIO (VI)

COTECA SAVOIA
Via Tartarotti, 48
ROVERETO (TN)

PROCELLI R.
Via F.lli Bronzetti, 29
38033 CAVALESE (TN)

COMPUTER POINT
Via Piovese, 37
35100 PADOVA

CAERT s.r.l.
Via Andora 11
35100 PADOVA

SIC ITALIA
Via Pillizzo, 23
35100 PADOVA

CAPUTO R.
Via S. Marco, 51/93
30100 VENEZIA

F.lli GIROTTO s.a.s.
Via Ospedale Provinciale, 25
SAN ARTEMIO (TV)

IRES s.p.s.
Via Dante
31040 CESSALTO (TV)

TALAMINI LIVIO e C. s.d.f.
Via Garibaldi, 2
31029 VITT. VENETO (TV)

FRIULI VENEZIA GIULIA

CATELLI FRANCO
P.za XX Settembre, 7
33170 PORDENONE

TREVISAN s.n.c.
P.za Duomo, 3
33100 UDINE

AUDIO MATRIX s.a.s. di ZINI
Via Marconi, 24
34133 TRIESTE

FORNIRAD di CASANELLI s.d.f.
Via Cologna, 10/D
34126 TRIESTE

EMILIA ROMAGNA

RECCHIONI Prof. ARTURO s.a.s.
V.le Oriani, 4
40137 BOLOGNA

BRIGHENTI F.lli s.n.c.
Via Riva Reno, 6
40122 BOLOGNA

COMPUTER SHOP
di RONCHI TONINO
Via Emilia, 199/B
40026 IMOLA (BO)

MICROINFORMATICS
P.zza dei Martiri, 31
41049 SASSUOLO (MO)

E.A.M. COMPUTER s.n.c.
Via C. Battisti, 17
42048 RUBIERA (RE)

PONGOLINI FELICE e F. s.d.f.
Via Cavour, 32
43036 FIDENZA (PR)

OVANI LINO
Via Meganazzi, 30
44022 COMACCHIO (FE)

ELECTRA s.r.l.
Via Copernico
47100 FORLÌ

CHIARI VITTORIO
Via Saffi, 49
47037 RIMINI (FO)

GIO.VE.
Via Polesine, 353
47023 CESENA

OTTICA ARCERI
di BONI MARIO e C. s.n.c.
Via Brenta, 4
47036 RICCIONE (FO)

TOSCANA

ACOUSTIC FIDELITY di BIAGIOTTI
Via Pisano, 161
50143 FIRENZE

RADIO TRADE CENTER s.r.l.
P.zza della Cervia, 22
55100 LUCCA

BENETTI IVO
Via Nazionale, 45
54039 TERRAROSSA (MS)

PUCCINI SILVANO
Via Cammeo, 64
56100 PISA

B.B.F. ELETTRONICA s.n.c.
Borgo della Vittoria, 18
51017 PESCIA (PT)

UMBRIA

M.T.E. di TEMPERINI ELETT. s.n.c.
Via XX Settembre, 76
06100 PERUGIA

MARCHE

ELETTROCASA di PIERONI
V.le Matteotti, 64
CIVITANOVA MARCHE (MC)

LA RO.FE di Rofini
Via Conce, 106
CORRIDONIA (MC)

AUDIO SHOP
Via Angelini, 68
ASCOLI PICENO

AULI ELETTRONICA
Via Turati
PORTO D'ASCOLI (AP)

DISCO CLUB
Via XXV Aprile
FERMO (AP)

DAVE ELETTRONICA
Via Formentini, 56
S. BENEDETTO DEL TRONTO (AP)

ani della Sirius Elettronica, ve proprio a tutto.

versi settori delle attività produttive: dal dentista all'albergo, dal negozio al teatro. E molti altri ancora.

Programmi semplici da usare, chiari, e razionali. E in italiano.

Tutto questo per valorizzare sempre di più Commodore 64.

GESTIONE LIBRERIE E BIBLIOTECHE	29.000
CASH COMPUTER RISTORANTI	89.000
MAILING LIST	75.000
WORD PROCESSOR	75.000
WORD ON MAIL	139.000
CONTABILITÀ FATTURE C/F	49.000
GESTIONE ALBERGHI	139.000
CASH COMPUTER PARRUCCHIERI	49.000
CASH COMPUTER GOMMISTI	49.000

MAGAZZINO E FATTURAZIONE	189.000
GESTIONE STUDI MEDICI	49.000
DUPLICATORE ARCHIVI	23.000
SINTETIZZATORE MUSICALE	29.000
DATA BASE	39.000
RUBRICA TELEFONICA	23.000
GESTIONE STUDI MEDICI ELABORATA	239.000
EXPERT SYSTEM GIOCO DIDATTICO	23.000

ovunque.

PASSI Hi-Fi
Via Galvani, 43
63023 FERMO (AP)

STEREO SOUND
Via Marina
CAMPOFILONE (AP)

PELLEGRINI MAURO
Strada Statale Adriatica
MARZOCCA DI SENIGALLIA (AN)

LAZIO

C.A.R.E.T. s.r.l.
L.go Belloni, 4
00191 ROMA

CIARLANTINI s.r.l.
Via Torrevecchia, 330
00168 ROMA

DISCOTECA FRATTINA
Via Frattina, 50
00187 ROMA

ELETTROPOKER di OLIVI MARIO
Via Volturno, 40/AB
00185 ROMA

FOTO FORNITURE DE BERNARDIS
P.za Cancelleria, 63
00186 ROMA

INTERPHOTO s.r.l.
Via Anastasio II, 438/440
00165 ROMA

NAUTICAL VIDEO COMPUTER GALLERY
Via Dogana Vecchia, 9/10
00185 ROMA

LEONARDO s.r.l.
Via Chopin, 29
00144 ROMA

METRO IMPORT
Via Donatello, 37
00196 ROMA

RADIO NOVELLI
P.le Prenestrina, 34
00176 ROMA

RAPPRESENTANZE E TECNICHE
Via Orazio, 24
00100 ROMA

R.P.M. di MANCINI
Via Giulia, 142 - 142/A - 143
00186 ROMA

TONEL CINE FOTO OTTICA RADIO TV
Porta Cavalleggeri, 15/1
00165 ROMA

PALOMBO ANTONIO
EQUIPE HI-FI
C.so Repubblica, 199/205
04100 LATINA

TOCCHINI TERESA
L.go Marconi, 14/16
040 11 APRILIA (LT)

V.V. VIDEO SOUND s.n.c.
Via Montegrappa, 34
GAETA (LT)

ABRUZZO E MOLISE

F.LLI COCOCETTA
Via Salluzzo, 91
L'AQUILA

ASPRON
Via America, 10
AVEZZANO (AQ)

VITTORIA NICOLA e C. s.a.s.
Via Privata P. Caledonia, 45
SULMONA (AQ)

NERI VINICIO
Via Nazzario Sauro, 80
GIULIANOVA (TE)

ALCINI DOMENICO
Via Nazionale
ROSETO DEGLI ABRUZZI (TE)

ELETTRONICA TERAMO
P.za Martiri Pennesi
TERAMO

FAGMA ELETTRONICA
V.le Regina Margherita, 74
PESCARA

STEFANO NICOLA
Via Madonna dell'Asilo
VASTO (CH)

PETROSEMOLO AMLETO
P.za Plebiscito, 40
LANCIANO (CH)

SISTEMA s.r.l.
Via Monsignor Bologna
CAMPOBASSO

F.LLI SCRASCIA
C.so Umberto I
TERMOLI (CB)

PUGLIA

ARTEL
Via Fanelli, 206/24
BARI

INSCAPE ADA BAGNARDI
Via Cardassi, 10/12
BARI

DE CESARE GIOVANNI
P.za Mazzini, 7/10
CISSERMINO (BA)

MCN di MITOLO
Via Labini, 34
BITONTO (BA)

BISCEGLIA Cav. DONATO
Corso Giannone, 235
S. MARCO IN LAMIS (FG)

MARANGI NICCOLI GRAZIA
Via A. Cagni, 27
BRINDISI

RUGGERO VITO
Via A. Sardelli, 75
S. VITO DEI NORMANNI (BR)

CISED di MARAGLINO MICHELE
C.so Roma, 199
MASSAFRA (TARANTO)

CAMEL di CAMASSA VINCENZO
Via Marimossi, 1/3
LECCE

MORCIANO FABIO
Via Madonna del Pane, 9
NOVOLI (LECCE)

CAMPANIA

AGNETI E AGNETO s.n.c.
Via Camillo Porzio, 79/87
NAPOLI

DI BIASE SALVATORE e Figli s.n.c.
Via Lanzieri, 15/25
NAPOLI

F.LLI DI PIETRO S.p.A.
P.za Mercato, 183
NAPOLI

NEW OTTICA
Galleria Umberto I, 55
NAPOLI

LUXOR RADIO
Galleria Umberto I, 51/54
NAPOLI

F.V. ELETTRONICA
di VACCHIANO FRANCESCO
Via Roma, 81
CICCIANO (NA)

DELLA FRONTE GAETANO
C.so Vittorio Emanuele, 140
AVELLINO

FLIP-FLOP s.r.l.
Via Appia, 68
AGRIPALDA (AVELLINO)

GRANDI MAGAZZINI PICCIOTTI
Via Madonna di Fatima, 15
SALERNO

CALABRIA

ANDREONI DI CALABRÒ FRANCESCO
Via Cairoli, 9
REGGIO CALABRIA

POLINESI ANNAROSA
Via Nazionale
ARCANI DI REGGIO CALABRIA

F.LLI GUALTIERI
C.so Mazzini, 157
COSENZA

GIORDANO GIOVANNI
P.za della Provincia, 4
COSENZA

AUDIO FIDELITY SHOP
di SQUILLACE
Via Spassari, 14
CATANZARO

GRANDINETTI RINALDO
Via Buccarelli, 9
CATANZARO

SICILIA

CFC s.r.l.
Via Notarbartolo, 2/R
PALERMO

F.C.F. S.p.A.
Via L. da Vinci, 238
PALERMO

MIGLIORE S.p.A.
Via Costantino, 43
PALERMO

PEDONE S.p.A.
V.le Strasburgo, 26
PALERMO

TRAPANI SALVATORE
Via Roma, 421
PALERMO

OTTICA LO VULLO
Via Atenea, 17
AGRIGENTO

PECORILLA IGINIO
Via Atenea, 24
AGRIGENTO

SCARPITTA ANTONINO
P.za Notai, 6
TRAPANI

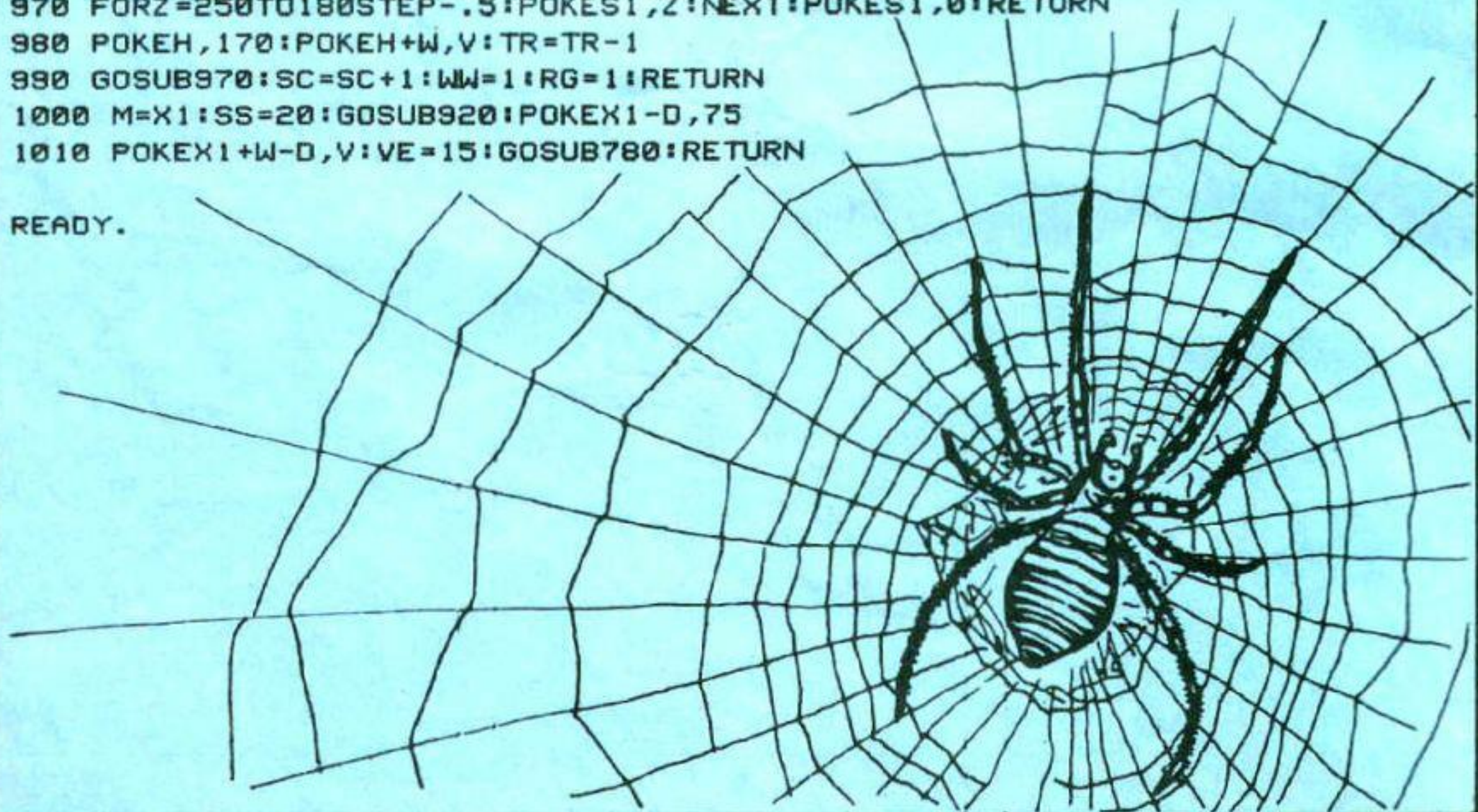
FERRAUTO s.r.l.
C.so Umberto, 233
CALTANISETTA

sirius elettronica

MILANOFIORI (ASSAGO) Palazzo F 2 - 5° piano - Tel.: 8245321

```
190 FORI=1TO21:PRINTX$;SPC(20)X$;:NEXT
200 FORI=8164TO8186:POKEI,102:POKEI+W,PU:NEXT:RETURN
210 R=10:C=3:GOSUB900:PRINT"CACCIA AL RAGNO"
220 PRINT"PREMI UN TASTO":GOSUB910
230 L=93:Y=L:X=7709:POKEX,Y:POKEX+W,V:S=22:N=32:M=32:U=42:VE=200
240 FORI=1TO8:GETA$:IFA$<>""THEN310
250 X=X+S:POKEX,U:POKEX+W,V:GOSUB920
260 GOSUB930:POKEX,L:POKEX+W,V:NEXT
270 IFL=NTHENGOSUB910:GOTO230
280 POKEX,U:POKEX+W,V:GOSUB930:IFPEEK(X+D)=32THENM=9
290 GOSUB930:POKEX+D,M:POKEX+W+D,PU:GOSUB940:GOSUB930:POKEX,N
300 S=-22:L=32:GOTO240
310 PRINT"":GOSUB180
320 FORG=1TO20:GETA$:NEXT
330 SP=0:RA=0:RR=0:WW=0:R=0:C=18:GOSUB900:PRINT""SC:PRINT""T
340 R=INT(13*RND(1)+5)
350 F$=CHR$(144)+CHR$(163)+CHR$(163)+CHR$(17)
360 F$=F$+CHR$(157)+CHR$(157)+CHR$(163)+CHR$(163)
370 F0$="        ":C=1:GOSUB900:PRINTF$:GOSUB910
380 VE=(4*RND(1)+.5)*30:C=INT(11*RND(1)+1)
390 X1=7710+C:POKEX1,Y:POKEX1+W,V
400 FORSS=1TO20
410 X1=X1+D:POKEX1,U:POKEX1+W,V
420 IFSP=0THENGETA$:IFA$=" "THENSP=1:GOTO460
430 IFA$="Q"THENPOKEV,0:POKES1,0:POKES2,0:POKES3,0:POKES4,0:END
440 GOSUB920:GOSUB930:POKEX1,Y:POKEX1+W,V:NEXT:POKEX1,U:POKEX1+W,V
450 GOTO580
460 T=T+1:X=7683+D*R:FORQ=1TO17:POKEX+Q,62:POKEX+W+Q,0:POKEX+Q,N
470 H=X+Q+1:IFPEEK(H)<>NTHEN490
480 NEXTQ
490 IFPEEK(H)=102ANDQ<7THENPOKEH,118:POKEH+W,PU:GOSUB950:PP=1
500 IFPP=1THENPOKEX1,Y:POKEX1+W,V:PP=0:NEXTSS
510 IFPEEK(H)=118THENNEXTQ
520 IFPEEK(H)=102THENGOSUB950:IFWW=0THENPOKEX1,Y:POKEX1+W,V:NEXTSS
530 IFRR=1THENGOSUB720:GOTO590
540 IFPEEK(H)=UTHENGOSUB980
550 IFRG=1THENRG=0:POKEX+Q+1,32
560 IFTR=0THEN630
570 IFPEEK(X+Q+1)=YTHENGOSUB940:RR=1:X2=X1:WW=1:GOSUB690:NEXTQ
580 IFPEEK(X1+D)=102THENGOSUB1000
590 IFPEEK(7773)=102THEN840
600 FORI=SSTO1STEP-1:X1=X1-D:POKEX1,N:TT=SIN(5):NEXT
610 C=1:GOSUB900:PRINTF0$
620 GOTO320
630 PRINT"HAI UCCISO TUTTI I"
640 PRINT"RAGNI IN";T;"TIRI !"
650 PRINT:PRINT"VUOI GIOCARE ANCORA ?"
660 GETA$:IFA$=""THEN660
670 IFA$<>"N"THENT=0:SC=0:TR=20:MM=0:GOTO310
680 END
```

```
690 POKEX+Q-21,74:POKEX+W+Q-21,V
700 IFPEEK(X+Q+23)=YTHENPOKEX+Q+23,85:POKEX+Q+W+23,V:RA=1
710 RETURN
720 IFRA=0THEN740
730 FORCC=7710+C+D*RTOX1-DSTEPD:POKECC,N:POKECC+W,V:NEXT
740 VE=15:IFSS=20THENGOSUB920:RETURN
750 POKEX1,N:FORM=X2TO(X2+(19-SS)*D)STEPD
760 POKEM,U:POKEM+W,V:GOSUB930:POKEM,N:NEXT
770 POKEM,U:POKEM+W,V:GOSUB920
780 POKEM,N:FORMM=MTO8150STEP-1:POKEMM,U:POKEMM+W,V
790 GOSUB930:GOSUB920:POKEMM,N
800 NEXTMM:FORA=1TO20:IFPEEK(MM-2)<>NTHENMM=MM-D:NEXTA
810 POKEMM-1,U:POKEMM+W-1,V:GOSUB940
820 POKEMM-2,102:POKEMM+W-2,PU:GOSUB910
830 POKEMM-1,N:RETURN
840 PRINT"▒▒▒HAI PERSO !!▒"
850 PRINT"▒HAI UCCISO";SC;"RAGNI"
860 PRINT"▒▒IN";T;"TIRI"
870 PRINT"▒▒MA";TR;"RAGNI SONO";:IFTR=1THENPRINT"▒▒▒▒▒▒▒D E'  ";
880 PRINT:PRINT"▒▒ANCORA VIVI";:IFTR=1THENPRINT"▒▒D"
890 GOTO650
900 RC$="▒"+LEFT$(R$,R)+LEFT$(C$,C):PRINTRC$;:RETURN
910 FORK=1TO1000:NEXT:RETURN
920 POKES3,230:FORZ=1TO10:NEXT:POKES3,0:RETURN
930 FORZZ=1TOVE:NEXT:RETURN
940 FORAA=200TO250STEP5:POKES3,AA:NEXTAA:POKES3,0:RETURN
950 FORAA=230TO128STEP-1.5:POKES4,AA:POKES1,AA
960 NEXTAA:POKES4,0:POKES1,0:RETURN
970 FORZ=250TO180STEP-.5:POKES1,Z:NEXT:POKES1,0:RETURN
980 POKEH,170:POKEH+W,V:TR=TR-1
990 GOSUB970:SC=SC+1:WW=1:RG=1:RETURN
1000 M=X1:SS=20:GOSUB920:POKEX1-D,75
1010 POKEX1+W-D,V:VE=15:GOSUB780:RETURN

READY.
```

GLI ALBERI BINARI DI RICERCA

Un breve listato che consente di rintracciare, dopo soli 16 tentativi, un certo dato inserito tra altri 65535!!!

Sicuramente tutti i fortunati possessori di un Personal-Computer operante in Basic e corredato da un floppy-disk si saranno almeno una volta posti il problema della memorizzazione e del trattamento di una serie di dati numerici-e-non (agenda telefonica, contabilità domestica, inventari, data-base etc.). Una grande maggioranza di questi, non avendo altre basi teoriche se non quelle acquisite in lunghe ore notturne passate leggendo riviste di informatica, hanno partorito programmi che, pur 'girando', non sono certo dei campioni di efficienza o di velocità.

Il problema che si erano posti, è infatti uno dei problemi più importanti nell'uso dei calcolatori (..'ordinateur' in francese..). Per questo motivo si è creata tutta una 'scienza', ad opera di eminenti informatici, per cercare di ottimizzare e sveltire tutte le operazioni da svolgere su una grande massa di dati quali la ricerca, la aggiunta, la cancellazione e l'ordinamento di dati. Una di queste teorie che mi accingo ad illustrare, corredata da programma, è quella che prende il nome di ALBERI BINARI DI RICERCA.

Questa teoria, affascinante un po' ostica, solitamente si affronta studiando seriamente il Pascal poichè gli alberi si 'sposano' particolarmente bene con un linguaggio strutturato quale il Pascal è.

Niklaus Wirth il "papà" del Pascal, (e non solo lui) ha dedicato a questo tipo di strutture, parte rilevante del suo libro "Algorithms + data structures = programs".

Ma cosa diavolo sono questi alberi, cosa c'entra la botanica nel mondo dell'informatica?

Come lo stesso Wirth li definisce, essi sono: una struttura dati che è naturalmente definita 'recursivamente' ed elaborata con algoritmi 'recursivi'. Un albero è costituito da un insieme finito di nodi ed è vuoto o costituito da un nodo (radice) con un numero finito di alberi disgiunti, chiamati "sottoalberi". Nessuno ha la pretesa (neanche lo stesso Wirth) che questa definizione possa chiarire le idee in merito.

Non volendo certo far ora un trattato sulla materia, (per un approfondimento dell'argomento vedi testo citato), limiterò la mia esposizione al solo caso particolare degli ALBERI BINARI. Tali alberi sono una semplificazione rispetto a quelli GENERALI poichè da ogni nodo non dipartono i nodi, ma solo e sempre 2 (fig. 1) (a) e (b). Quella parola strana che troviamo nella definizione, 'recursivo', 'recursivamente', etc. è sicuramente sconosciuta a tutti coloro che non hanno mai lavorato su altri linguaggi se non il Basic. Linguaggi 'strutturati' come il Pascal infatti permettono l'implementazione di quella tecnica potentissima, seppur di non immediata comprensione, che è la RICORSIONE.

Il Basic (non permettendo altro che l'implementazione di procedure 'iterattive'), si è precluso per sempre l'uso degli alberi binari? Per fortuna no, e il programma che presento, infatti, ne è la prova.

Vediamo di chiarire ora un altro punto importantissimo: a cosa servono gli alberi, o più precisamente i 'search binary trees' (alberi binari di ricerca). Questi alberi espletano tutte la loro potenza nella memorizzazione e nell'aggiornamento di una grande massa di dati dello stesso tipo. Tali dati possono essere singoli (esempio: un numero) o composti (esempio: record persona = cognome, nome, età, sesso). Per fare un esempio concreto di utilizzo, pensiamo di immettere nel nostro calcolatore l'elenco degli abbonati al telefono. Se non usiamo gli alberi, siamo costretti ad usare un array (o file) sequenziale nel quale aggiungere (in coda) o togliere elementi. I problemi però sorgono al momento dell'ordinamento per ordine alfabetico, nella ricerca di un dato numero etc.

Con gli alberi tutti questi problemi sono risolti nel migliore dei modi: grazie alla sua struttura 'bidimensionale' teoricamente infinita (alle 'foglie' inferiori possono sempre essere aggiunti altri 'nodi'), al momento dell'immissione, il nuovo dato viene automaticamente posto in ordine alfabetico e, durante la ricerca, nel peggiore dei casi, il dato voluto è trovato in LOG2 (n) tentativo. Esempio: avendo 65535 dati memorizzati, il dato voluto sarà trovato al massimo in 16 tentativi!

C'è ora da far una importante precisazione; nel caso di dati complessi, come può essere il caso della agenda telefonica,

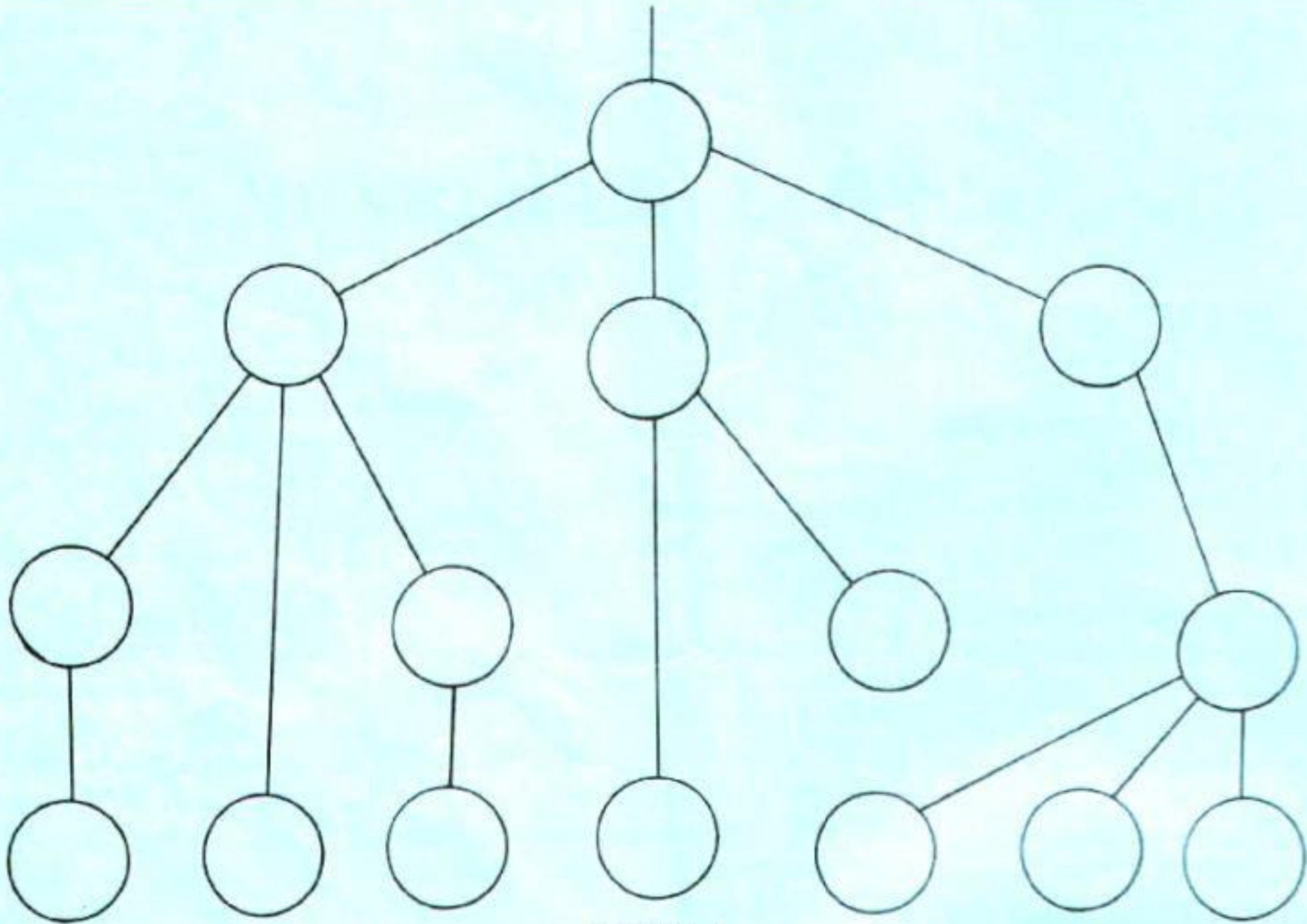



FIGURA 1 - a
Albero generale.

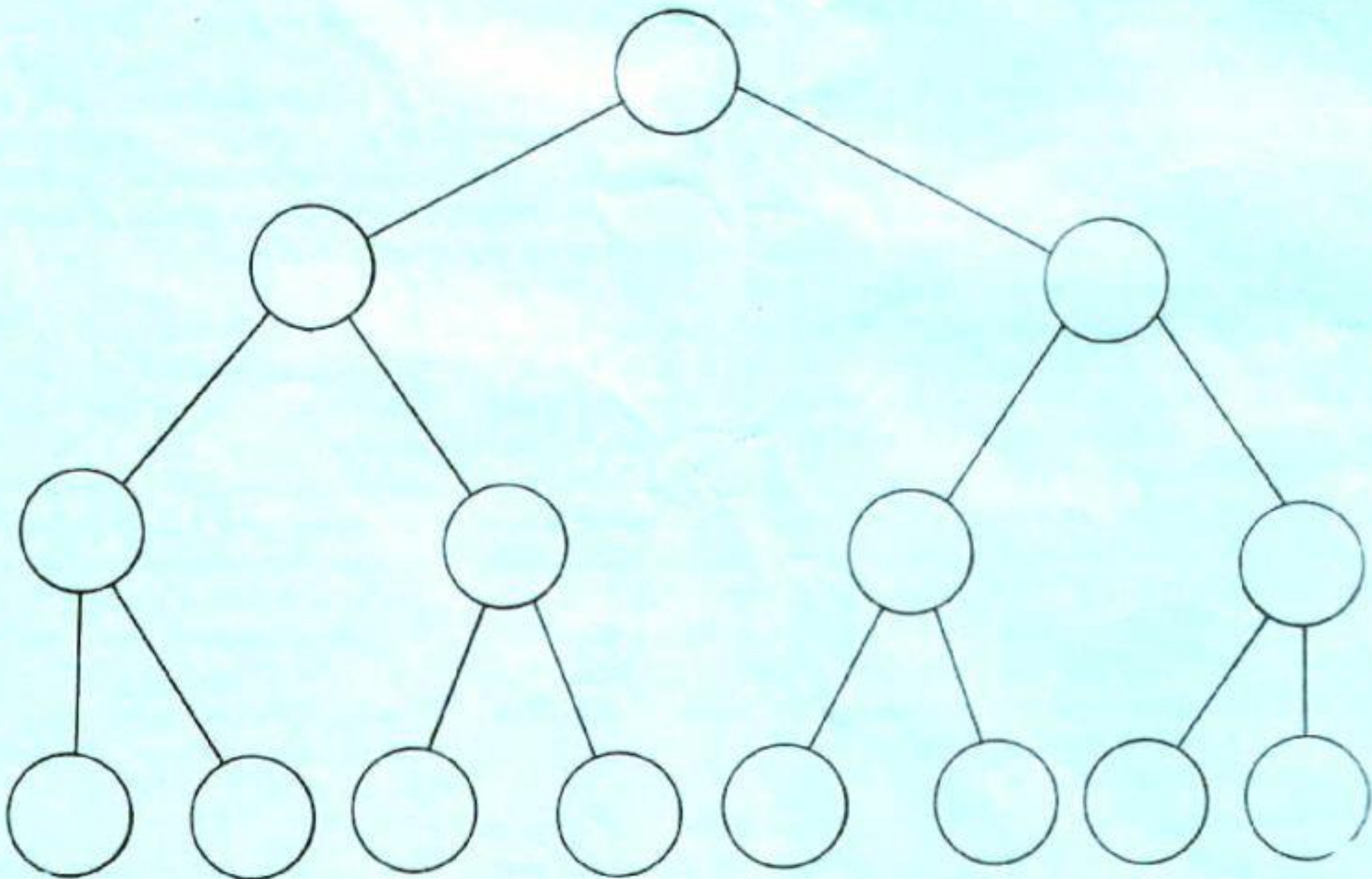



FIGURA 1 - b
Albero binario

il dato stesso è diviso in vari "campi" (cognome, nome, etc), uno solo dei quali però costituirà la "chiave" con la quale si svolgeranno tutte le operazioni. Se nell'esempio precedente la chiave è il campo 'cognome', allora tutti i dati saranno posti secondo l'ordine alfabetico dei cognomi, si potrà ricercare un certo cognome oppure cancellarlo e così via. A quel dato cognome però saranno legati indissolubilmente gli altri dati a lui relativi quali il nome, l'età, etc, sebbene su questi non venga svolta nessuna operazione inerente all'albero.

Dalla teoria scendiamo più nella pratica e vediamo di analizzare quelle idee che hanno permesso la realizzazione del programma.

FIGURA 2

Il programma

Vediamo di analizzare le differenze sostanziali tra il Pascal ed il Basic a causa delle quali saremo costretti a porre delle semplificazioni strutturali. (Purtroppo tutte queste differenze sono a svantaggio del Basic). In Basic è impossibile una allocazione 'dinamica' di dati in memoria, non si è in grado cioè di riservare uno spazio nella memoria centrale per un certo dato (singolo o composto che sia) e rimandarne l'indirizzo di locazione (hex) in una variabile. Per sopperire a questa 'deficienza' siamo costretti ad utilizzare un array per memorizzare i nostri dati, imponendo purtroppo a priori una limitazione alla struttura teorica dell'albero: si deve definire a priori il numero massimo di elementi che essi può contenere, provocando così un grande spreco di memoria per tutto il tempo che ci saranno pochi dati, e impedendo assolutamente l'ingresso dell'ennesimo +1 dato. Tale numero massimo sarà contenuto nella variabile MAX%: cambiandolo si potrà così 'personalizzare' a piacimento il programma.

chiave	punt. sinistro	punt. destro
A$ (0) = PAOLO	PT% (0,0) = 1	PT% (0,1) = 2
A$ (1) = GIULIANA	PT% (1,0) = -1	PT% (1,1) = 3
A$ (2) = ROSANNA	PT% (2,0) = -1	PT% (2,1) = -1
A$ (3) = LUIGI	PT% (3,0) = -1	PT% (3,1) = -1

TABELLA I
La allocazione dei dati (chiave e puntatori) nella memoria del calcolatore.

Altra deficienza del Basic, sia detto per inciso, è quella di non poter 'passare i parametri' alle subroutine per 'referenza', il che comporta artifici non indifferenti ed una certa perdita di tempo e di chiarezza. Se quindi A$T è un array che contiene i dati da memorizzare sappiamo che deve essere 'lungo' MAX% ma non abbiamo ancora detto niente sulle sue dimensioni. Esse infatti dipendono strettamente dalla natura dei dati da memorizzare. Se i dati sono singoli e di natura numerica, l'array risulterà definito DIM A (MAX%). Se invece i dati saranno alfanumerici e magari composti (esempio: cognome, nome, indirizzo, n.ro telefono), allora bisognerà dimensionare il vettore diversamente: DIM A$ (MAX%.,3).

Ma allora è sempre la stessa storia!? Noo!!! Di contorno a questo array infatti ne esistono altri ed uno in particolare che fa sì che si possa parlare di ALBERI: è l'array PT% (MAX%.,1). Questo, la cui natura è rivelata dal segno %, contiene i 'puntatori' destri e sinistri di ogni nodo. Mamma mia, qui si entra nel difficile! Come si è potuto vedere in figura 1, ogni elemento è legato al nodo 'padre' tramite delle linee chiamate 'link'; esse sono in realtà delle 'informazioni' residenti nel nodo padre e contenenti l'indirizzo (l'indice dell'array A) nel quale il nodo figlio è presente. Risulta così evidente che l'accesso ad ogni nodo può avvenire solamente attraverso un cammino 'discendente', partendo dal primo nodo e seguendo di volta in volta il cammino a destra o a sinistra (secondo le necessità) indicato dai 'puntatori'.

COMMODORE SCHOOL

Nell'esempio in figura 2, il primo nodo contenente il dato 'PAOLO' e allocato in A$(0), sa che alla sua'sinistra' c'è il dato contenuto in A$ il cui indice è contenuto nel suo 'puntatore sinistro' PT%(0,0) = 1, cioè A$(1) = GIULIANA. Lo stesso per la sua destra PT%(0,1) = 2, A$(2) = ROSANNA etc. Se un nodo non ha discendenza destra o sinistra, i suoi relativi puntatori dovranno assumere un valore particolare detto NIL. Nel programma questo valore è stato scelto -1 poichè indice 'impossibile' di un array. Il problema ora è però capire come aggiornare automaticamente questi puntatori durante l'immissione di nuovi dati, e con che criteri.

L'immissione dei dati e l'aggiornamento dei puntatori

La routine INGRESSO è stata realizzata rendendo iterattiva la corrispondente e 'celeberrima' procedura ricorsiva suggerita da N. Wirth. Innanzitutto diciamo qual è il criterio che permette la costruzione del nostro albero. Al momento dell'ingresso il nodo deve essere posto alla sinistra di un nodo con dato 'maggiore' (nel caso alfanumerico si intende l'ordine alfabetico) e alla destra di uno con dato 'minore' già presente. N.B. Da ciò risulta evidente che *non è possibile* inserire due dati uguali. L'indirizzo del primo nodo è contenuto in una variabile RT%.

Supponiamo nel caso di figura 2 di dover inserire un nuovo dato contenente il nome FEDERICA posto in una variabile temporanea X$. Prima dell'inserimento, la situazione dei dati in memoria, è visibile in tabella 1. Sia P% un puntatore variabile che permette di 'scendere' attraverso l'albero. Tale puntatore viene inizializzato (riga 70) a puntare al primo nodo (radice) dell'albero, PAOLO in questo caso. Sul nodo puntato viene svolto il confronto determinante (righe 220-230): si decide, tramite gli operatori relazionali < e > (anche tra stringhe per fortuna!), se X$ deve essere posto alla destra o alla sinistra del nodo puntato da P%. Poichè in questo caso X$ < A$(P%) (FEDERICA < PAOLO), allora P% diventa uguale al puntatore sinistro del nodo puntato (P% = PT%(P%,0)) e il procedimento si ripete; si dice che P% è "sceso a sinistra". P% ora punta a GIULIANA e siccome FEDERICA è ancora 'minore', P% viene di nuovo aggiornato. Così facendo però, P% viene ad assumere il valore NIL. Il procedimento di 'discesa' ha così termine (riga 200): si è cioè constatato che il dato FEDERICA non esisteva già, e se ne è trovato il posto ove locarlo.

Soltanto ora il dato è inserito (riga 250) nell'array A$ nella prima casella vuota (tratterò in seguito il criterio con il quale tale casella è trovata), il cui indice deve essere memorizzato nel relativo nodo-padre (GIULIANA in questo caso). Solo quando il dato è memorizzato in A, ed i puntatori sono aggiornati, si potrà parlare di "inserimento del dato nell'albero". Siccome, come ho già detto, il percorso in un albero può solo essere 'discendente', l'indirizzo del nodo padre, perso inevitabilmente da P%, è tenuto da un altro puntatore, ausiliario NX% che è sfalsato di un nodo rispetto a P%. Per sapere poi se tale nodo padre era < o > di X$ (anche tale informazione era stata persa), ci pensa un 'flag' apposito: ST%. Ad ogni 'chiamata', questa procedura immette il dato nell'albero, aggiornandolo.

Vorrei sottolineare che è questo uno dei fattori più importanti della teoria degli alberi: il calcolatore mette in ordine il dato al momento della digitazione ed il tempo per far ciò risulta irrisorio rispetto a quello impiegato da una persona a battere sui tasti. Questa perdita di tempo e quindi praticamente inesistente, mentre al momento della estrazione in ordine dei dati, si assiste a dei prodigi di velocità. Problema in un certo senso simile dal punto di vista algoritmico, seppur nettamente più semplice, è quello della ricerca di un certo dato in memoria.

La ricerca di un dato nell'albero

Sia sempre X$ la variabile temporanea contenente questa volta la 'chiave' del dato da ricercare. Riferendoci al nostro albero appena aggiornato di fig. 2, sia da trovare la 'chiave' LUIGI. Dopo aver come al solito inizializzato P% a puntare alla radice, vengono di nuovi effettuati i confronti relazionali (righe 330-340) simili a quelli svolti nella procedura INGRESSO. In base a questi, P% scende lungo l'albero dimezzando ogni volta il numero di posti ove cercare il dato (..ricerca binaria..), viene così spiegata, l'enorme velocità di ricerca ed il nome stesso di questi tipi di alberi. Scendendo così, con lo stesso criterio con il quale il dato era stato immesso in memoria, si giungerà per forza con il trovarlo se esiste; se si ricerca un dato non presente, la 'discesa' di P% verrà bloccata da un inequivocabile valore NIL. In questo caso l'inesistenza della chiave cercata viene comunicata (riga 320).

Nel caso invece che il dato sia stato trovato, esso apparirà sullo schermo, preceduto (a puro titolo informativo) dal numero della casella da lui occupata nell'array A$, e seguito da due numeri, rispettivamente i puntatori sinistro e destro.

Riferendoci all'esempio:

- LUIGI < PAOLO quindi scendi a sinistra
- LUIGI > GIULIANA quindi scendi a destra
- LUIGI = LUIGI quindi stampa:

```
3 LUIGI     -1     -1
```

Continuando con le procedure INGRESSO e CERCA, il nostro albero può così essere creato e 'visitato'; altre routine però sono necessarie per completare questa teoria. Una di esse è quella tramite la quale si possono estrarre tutti i nodi in ordine.

Estrazione in ordine delle chiavi

Supponiamo di aver ampliato il nostro albero con l'aggiunta di altre chiavi. Sia l'albero risultante quello in figura 3. Pos-

siamo a prima vista veder che l'ordine alfabetico di quei nomi è:
FEDERICA - GIULIANA - LUIGI - PAOLO - RENZA - ROSANNA

Pensare un metodo iterattivo (ricorsivamente è facilissimo) che possa fare il cammino per estrarre in ordine tutte le chiavi (e tutti i dati a lei legati) è stata la cosa più difficile che ho dovuto risolvere. Per poter meglio spiegare il procedimento adottato, si veda il flow-chart di figura 4. Ho dovuto ricorrere purtroppo a due array d'appoggio, o meglio di un array bidimensionale PR%(MAX%,1). Lo scopo di questo array è di tenere per ogni nodo un 'flag' di bloccaggio e l'ordine con il quale i nodi sono visitati. Questo array contenente i flag deve essere inizializzato a contenere tutti 0. Tale inizializzazione è svolta dalle righe 550-560.

Tornando alla procedura, leggendo il flow-chart e sapendo che all'inizio tutti i nodi non sono 'bloccati', vediamo che P% scende al nodo più in basso a sinistra, cioè a quel nodo che contiene il dato 'minore' di tutti gli altri secondo il principio di inserimento. Raggiuntolo, e non avendo questi sottoalberi a sinistra, il calcolatore deve eseguire l'OPERAZIONE che si vuole fare su ogni nodo visitato, nel nostro caso la stampa del dato stesso (riga 590).

Si guarda poi se il nodo ha sottoalberi a destra. In caso affermativo si 'scende' a destra, altrimenti, si "risale" al nodo precedente tramite l'uso dell'array PR% (riga 620). Se questo nodo precedente ha il flag di bloccaggio settato o non ha sottoalberi a sinsitra (se non può andare a sinistra in parole povere), allora si risale ancora al precedente (riga 630). N.B. Precedente qui è inteso non nell'ordine numerico o alfanumerico delle chiavi, ma relativamente all'ordine con il quale i nodi vengono visitati durante l'esecuzione della procedura.

Durante questo processo di 'risalita' appena viene incontrato un nodo non

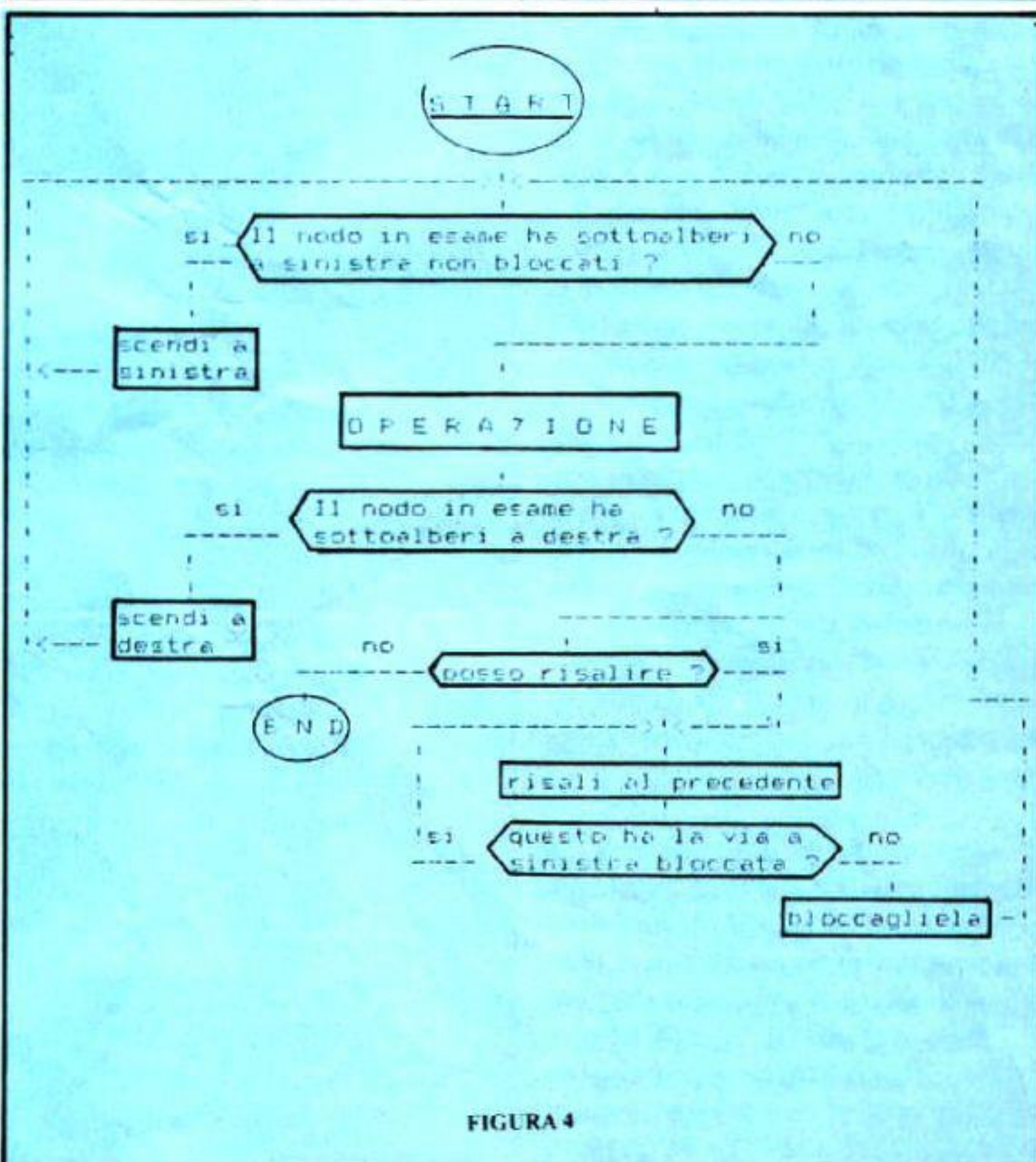
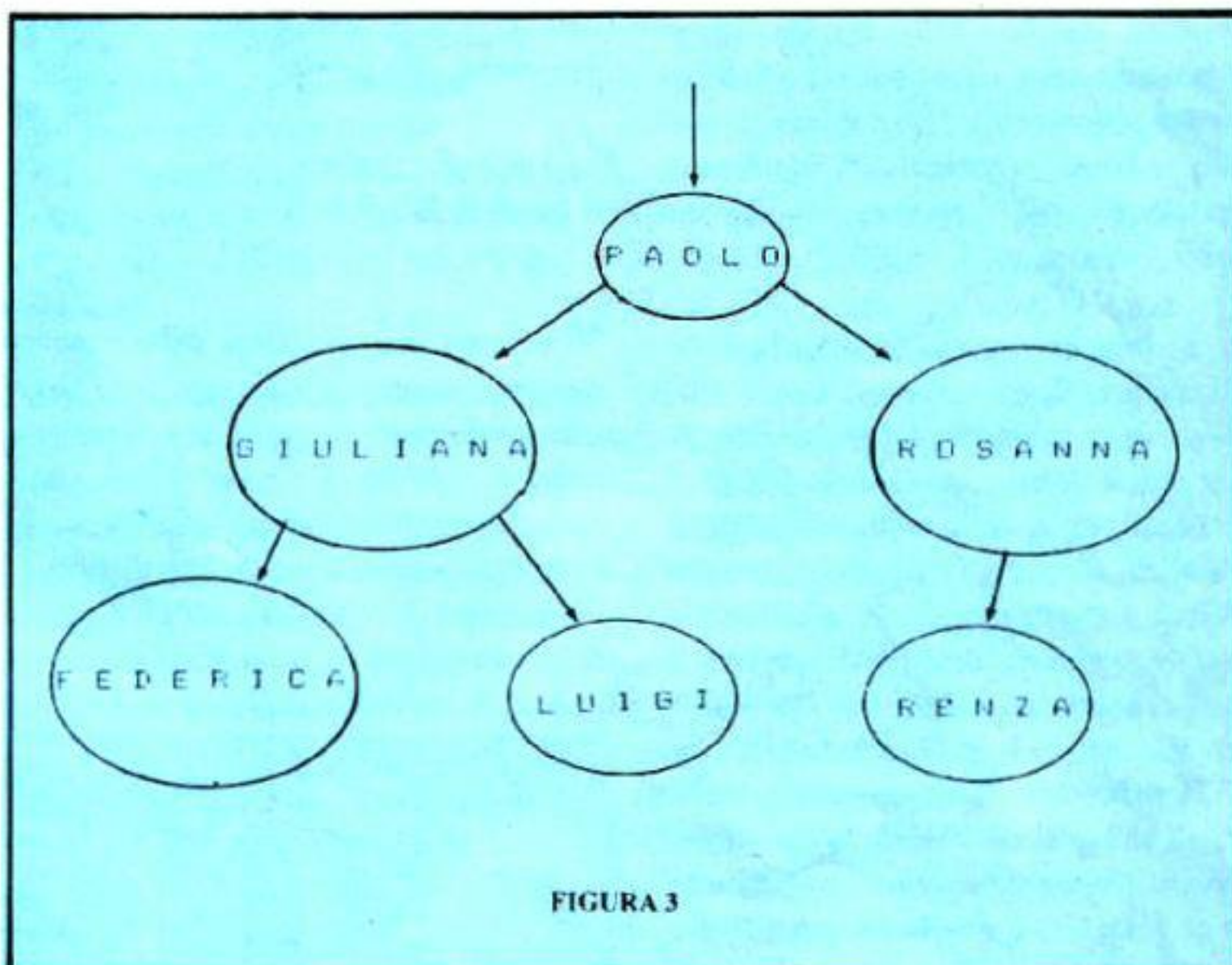



FIGURA 3

FIGURA 4

bloccato, esso viene bloccato (riga 640) ed il procedimento riprende. Lo STOP di questa procedura è dato dal fatto di terminare i nodi 'precedenti': il contatore di precedenza, CP%, diventa cioè -1 (riga 610).

Facendo riferimento alla figura 3 vediamo in 'trasparenza' cosa succede. Viene preso in esame PAOLO, avendo discendenza sinistra, si scende a GIULIANA e poi per lo stesso motivo a FEDERICA. Non avendo questo nodo sottoalberi sinistri, esso viene stampato. FEDERICA non ha sottoalberi destri, per cui si risale al precedente (GIULIANA). Quest'ultimo non avendo la via a sinistra bloccata, gli viene bloccata. Si riconsidera quindi lo stesso nodo che avendo ora la via sinistra bloccata, viene stampato. Contrariamente al precedente, questo nodo ha discendenza a destra, dove si scenderà. Verrà quindi stampato LUIGI poichè privo di sottoalbero sinistro; essendo però questi anche privo di quello destro, si risale a GIULIANA e, siccome questo ha la via sinistra bloccata, si risale ancora a PAOLO e gli si blocca la via a sinistra. Per questa ragione alla successiva passata PAOLO verrà stampato e si prenderà in considerazione il suo nodo a destra (ROSANNA) e successivamente quello a sinistra (RENZA) che verrà stampato. Si risalirà quindi a ROSANNA e se gliene bloccherà la via sinistra cosa che ne provocherà la stampa alla successiva passata. A questo punto si risalirà a PAOLO ma avendo questi la via a sinistra bloccata, si tenterà di risalire ancora, ma 'sopra' non esiste niente, per cui il processo ha termine. Uff, infine!! C'è da perdersi in mezzo a queste arborescenze computerizzate.

Facezie a parte il programma ha estratto in ordine alfabetico tutti i nodi immessi nell'albero c.v.d. A puro titolo informativo dirò che questo tipo di 'visita' ad un albero binario, viene chiamata di tipo INORDER (SINISTRA - RADICE- DESTRA), che si contrappone ad altri 2 tipi di visita detti rispettivamente PREORDER (RADICE - SINISTRA - DESTRA), e POSTORDER (SINISTRA - DESTRA - RADICE).

La cancellazione di nodi dell'albero

Vediamo ora all'ultima delle routine che costituiscono il package presentato sulla teoria degli alberi. E' questa chiaramente la procedura di cancellazione dei nodi. A prescindere dal linguaggio in cui si opera, questo problema risulta essere il più pepato. Per rendere l'idea si pensi all'albero di fig. 3 e pensate di dover togliere il dato PAOLO; come pensate che risulti l'albero dopo tale 'potatura'? Tenete presente che l'ordine intrinseco non deve venir mutato. Per fortuna anche questo problema lo ha risolto il sig. Wirth una volta per tutte.

Cominciamo con l'analizzare la casistica:

- si deve cancellare una foglia;
- si deve cancellare un nodo con un solo sottoalbero;
- si deve cancellare un nodo con due sottoalberi.

Nel primo caso la soluzione è banale: si recide il link nel nodo padre (righe 450-

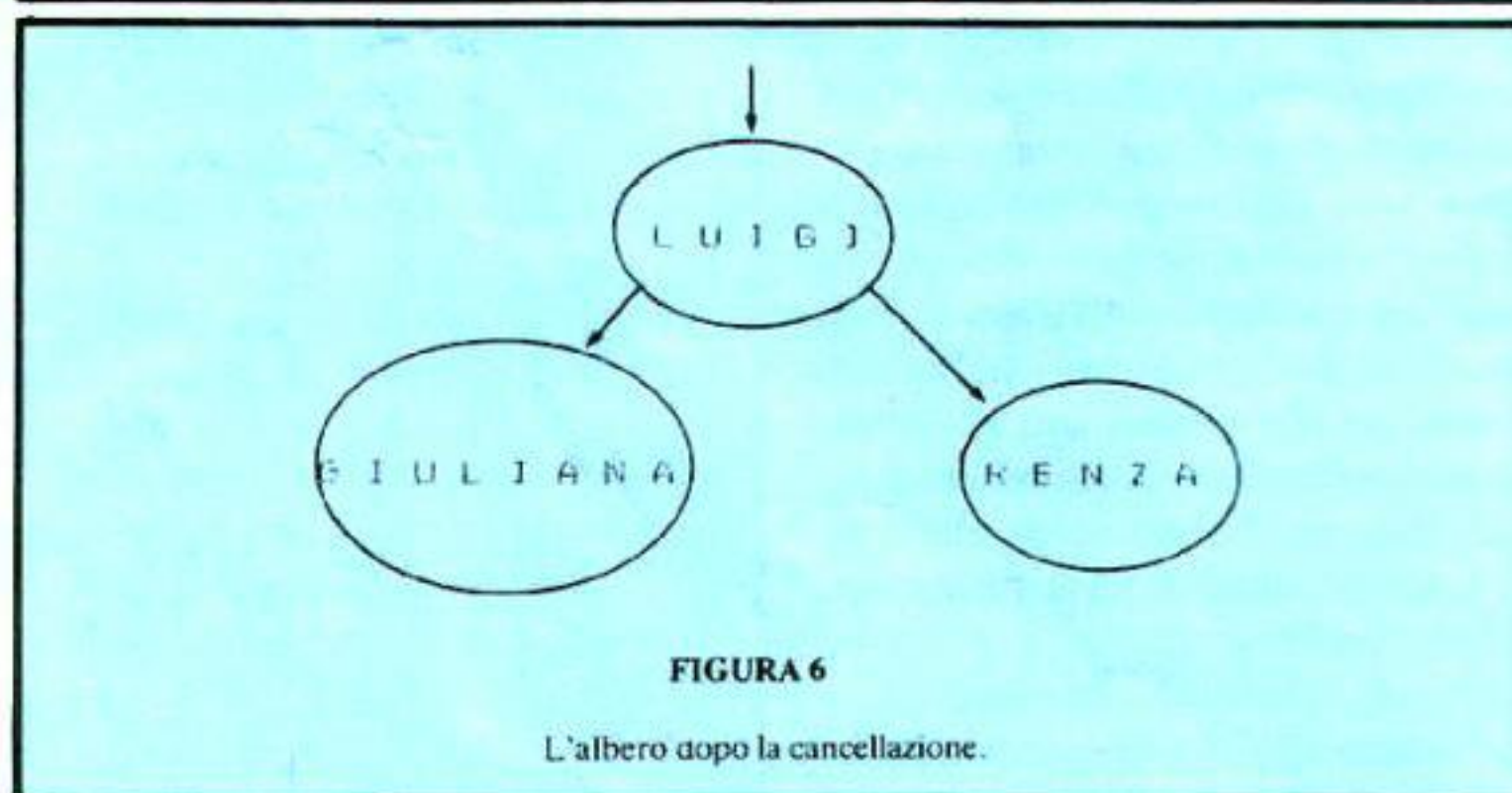
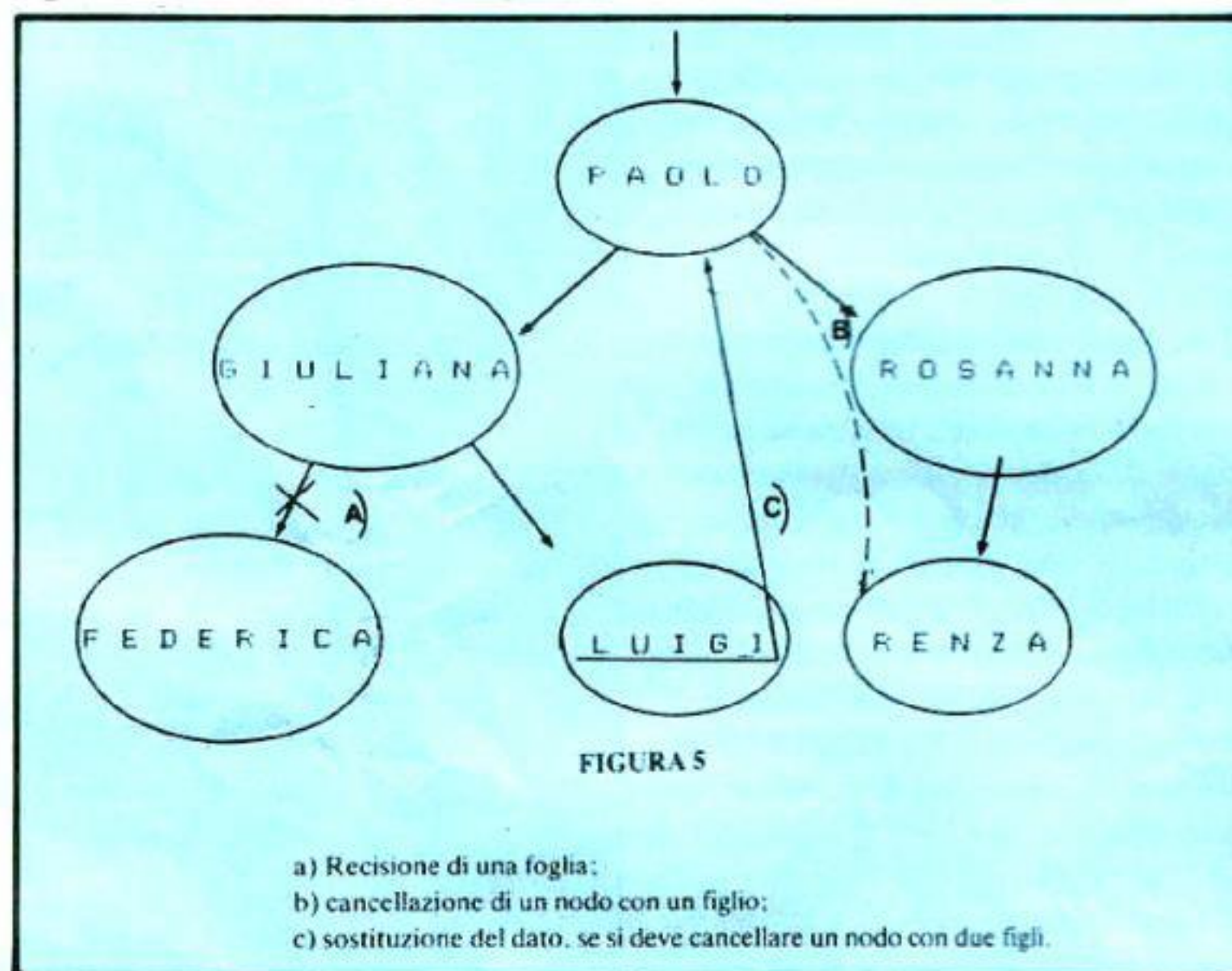
PAOLO
GIULIANA
ROSANNA
FEDERICA
LUIGI
RENZA
A)
B
C)

FIGURA 5

a) Recisione di una foglia;
b) cancellazione di un nodo con un figlio;
c) sostituzione del dato, se si deve cancellare un nodo con due figli.

LUIGI
GIULIANA
RENZA

FIGURA 6

L'albero dopo la cancellazione.

460) ed essendo così il dato irreperibile, risulterà cancellato (fig. 5-a). Esempio: per cancellare FEDERICA basta porre il puntatore sinistro di GIULIANA uguale a NIL.

Anche nel secondo caso le cose non sono troppo difficili poichè basta aggiornare il puntatore del 'padre' perchè punti al proprio 'figlio' (nuovamente righe 450-460). Esempio: si debba cancellare ROSANNA, allora basterà porre il puntatore destro di PAOLO uguale al puntatore sinistro di ROSANNA e così risulterà che PAOLO avrà come figlio destro RENZA (fig. 5-b). Anche in questo caso il nodo ROSANNA non scomparirà effettivamente dalla memoria centrale, ma risulterà irreperibile nell'ambito dell'albero.

Come già accennato, i problemi insorgono nel terzo caso, quando cioè il nodo da cancellare ha doppia 'prole'... sti figli!! Innanzitutto bisogna, come anche nei casi precedenti, trovare il nodo da cancellare (righe 390-400). Entrano in gioco a questo punto altri due puntatori, ausiliari Q% e Q1%. La situazione è dunque questa: P% punta al nodo da cancellare, mentre Q% e Q1% vengono fatti 'scendere' sfalsati di uno, (riga 480) al nodo più a destra nel sottoalbero di sinistra di quello da cancellare. AAAGHH!! che COMA! Un attimo di pausa, andate a prendere un caffè e poi continuiamo stoicamente. Per fissare le idee supponiamo di dover cancellare PAOLO. A lui punta P%, poi facciamo scendere Q% e Q1%, in modo che il primo vada a puntare a LUIGI e l'altro a GIULIANA. LUIGI è infatti il nodo più a destra del sottoalbero sinistro partendo da PAOLO. Chiaro ora?

Perchè, comunque, tutto queste complicazioni? Semplice, il nodo così trovato è quello che immediatamente precede il dato da cancellare nell'ordine (numerico o alfabetico) con il quale i dati sono stati immessi. Si svolge a questo punto il trasferimento del dato contenuto nel nodo puntato da Q% a quello puntato da P% (riga 490) (fig. 5-c). Il vecchio nodo che conteneva LUIGI, rientrando in uno dei casi precedenti, può essere facilmente cancellato. Dopo questa 'potatura', il nostro albero sarà quello che compare in figura 6. Resta a questo punto un'ultimo argomento da affrontare che riguarda strettamente questo programma: l'utilizzo della memoria.

L'utilizzazione della memoria

Come ho già spiegato siamo stati costretti ad utilizzare un irridimensionabile array per contenere i nastri dati, il che è una non lieve minaccia alla flessibilità dell'algoritmo. Se aggiungiamo poi come funziona la routine di cancellazione, vediamo che nel giro di poco tempo si arriverebbe al completamento dello spazio disponibile, avendo, al limite pochi dati immessi.

Ho quindi voluto aggiungere un piccolo 'trucco' che mi permetta di utilizzare anche quei nodi che, pur non cancellati fisicamente, sono stati recisi dall'albero. Ciò che ho fatto è, se vogliamo, l'uovo di colombo, poichè tengo conto in una lista, LS%(MAX%), dei nodi cancellati (righe 510-530). Grazie ad un contatore di nodi presenti in questa lista, LC%, al momento dell'immissione di un nuovo dato, posso sapere se esiste dello spazio riutilizzabile; in tal caso CT%, che mi dice dove memorizzare i' nuovo dato, sarà caricato con questo indirizzo di memoria disponibile. In caso contrario, quando cioè non è ancora stato cancellato nessun nodo, CT% viene caricato con un valore, presente in C%, che fornisce semplicemente l'indirizzo della "successiva" casella disponibile dell'array A (riga 190). Vediamo ora la parte più pratica e quindi più interessante: come «personalizzare» il programma.

La personalizzazione del programma

Come avrete potuto constatare, ciò che fa il programma che presento è di una stupidità agghiacciante. Mio unico scopo era infatti di presentare le routines "generali" per il trattamento degli alberi binari in Basic. L'importante, ora che si è capito il 'nocciolo' della questione, è di poter utilizzare secondo le proprie necessità questi algoritmi. Innanzittutto bisogna stabilire quanti elementi al massimo si vogliono memorizzare, e caricare tale cifra in MAX% alla riga 20. Avendo ben chiaro che tipi di dati memorizzare, adeguate il dimensionamento dell'array A$. E' chiaro che cambiando la dichiarazione di A$ bisogna adeguare in tutto il resto del programma le diciture che interessano tale variabile.

Prima di richiamare ciascuna delle routine, fare eseguire le istruzioni contenute alla riga 70. Nella routine INGRESSO, alla riga 250, immettere tutti i dati relativi ai campi costituenti il nodo A (oltre a X utilizzare altre variabili di appoggio), e mantenere le due ultime istruzioni presenti. Esempio.

```
250 A$(CT%,0) = X$ : A$(CT%,1) =
X1$ : < etc >:
PT%(CT%,O) = NIL% : PT%(CT,1) =
NIL%
```

Nella routine CERCA, far eseguire l'operazione voluta sul nodo cercato, alla riga 350. Nella routine INORDER, far eseguire l'operazione voluta sui nodi estratti, alla riga 590. Nella procedura cancellazione, trasferire tutte le informazioni contenute nel nodo puntato da Q% a quello puntato da P%, alla riga 490.

Esempio:

```
490 A$(P%,0) = A$(Q%,0) : A$(P%,1) =
A$(Q%,1) : π < etc >.
```

A questo punto è bene ribadire un concetto, per altro già accennato: nel caso di dati composti da più «campi» (esempio: cognome, nome, indirizzo, età), l'ordinamento, la ricerca etc., può essere fatta solo relativamente ad UN SOLO CAMPO. Nel caso precedente per esempio, posso scegliere di creare l'albero ordinato per cognome. Il campo-chiave può essere qualsiasi, ma generalmente si usa il primo. Avendo il programma utilizzato un solo campo che è per forza anche campo chiave, ogni qual volta che nel listato si

COMMODORE SCHOOL

incontra la variabile A$, quello è il posto per utilizzare il vostro campo chiave. Per intenderci, se in un'istruzione trovate A$(P%) e avete più campi di cui il primo "chiave", dovrete sostituire la dicitura in A$(P%,O). Le righe 60-140, sono quello che in gergo è chiamato il "main program", che contiene cioè tutte le «chiamate» alle varie procedure. E' logico che questo deve essere fatto su misura per i vostri scopi; esso non ha come si può notare, nessuna influenza sulle routines presentate.

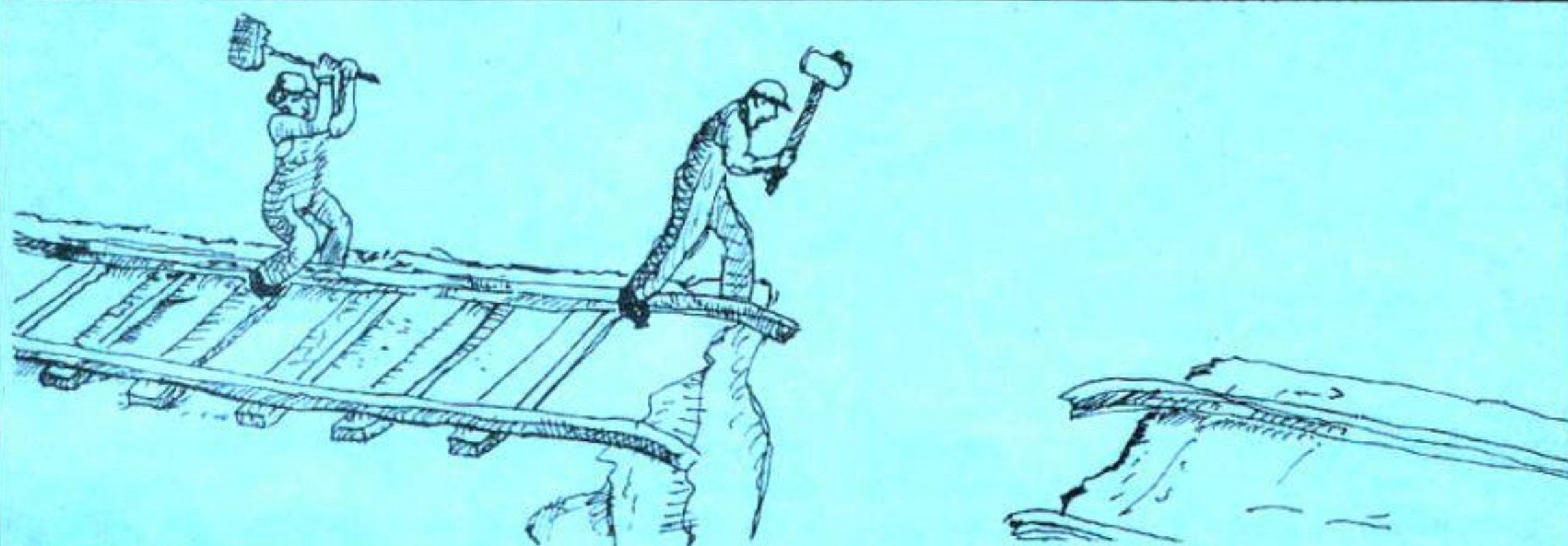
Siamo infine giunti alla fine di questa faticaccia ed è buffo notare che per «spiegare» gli alberi ad un calcolatore (mero ammasso di circuiterie elettromeccaniche) bastano poche decine di succinte istruzioni, mentre per gli uomini (intelligentissimo padrone e dominatore del Mondo) sono occorse parecchie pagine e ore di duro lavoro. Meditate gente, meditate.

Paolo Caprioglio
Corso Francia, 224
10146 Torino - Tel. 011/792023

Uso del programma

Riporto qui il listato di un esempio di utilizzazione del programma. Tramite il comando **I viene chiesta la chiave e il dato viene inserito nell'albero. Vengono quindi immessi 6 dati in un ordine del tutto casuale. Tramite **S l'albero viene visitato e tutti i dati sono estratti in ordine alfabetico.

Digitando poi ** R si può ricercare una data chiave.

```
COMAND--> **I
DIMMI UN NOME PAOLO
COMAND--> **I
DIMMI UN NOME GIULIANA
COMAND--> **I
DIMMI UN NOME FEDERICA
COMAND--. **I
DIMMI UN NOME ROSANNA
COMAND--. **I
DIMMI UN NOME RENZA
COMAND--> **I
DIMMI UN NOME LUIGI
COMAND--. **S
FEDERICA
GIULIANA
LUIGI
PAOLO
RENZA
ROSANNA
COMAND--> **R
CHIAVE GIULIANA
1 GIULIANA        2              5
COMAND--> **D
DIMMI LA CHIAVE GIULIANA
COMAND--> **R
CHIAVE GIULIANA
CHIAVE INESISTENTE
COMAND--> **I
DIMMI UN NOME GIULIANA
COMAND--> **R
CHIAVE GIULIANA
2 GIULIANA        -1             -1
COMAND--> **S
FEDERICA
GIULIANA
LUIGI
PAOLO
RENZA
ROSANNA
COMAND--> **I
DIMMI UN NOME GIULIANA
IL NODO ESISTE GIA'
COMAND--> **.
```

```
1 REM  ALBERI BINARI
2 REM  PER TUTTI I COMMODORE CON ALMENO 32K DI MEMORIA
3 REM  DI CAPRIOGLIO PAOLO
4 REM  CORSO FRANCIA 224
5 REM  10146 TORINO  T. 011/792023
7 REM  REVISIONE DI MARIANI GIANCARLO
8 REM  TEL. 0362/72565
9 POKE53280,1:POKE53281,1:REM SOLO PER C64
10 PRINT"■":OPEN1,0
15 REM *** INITIALIZE ***
20 MAX%=100:NIL%=-1:F%=-1
30 RT%=NIL%:C%=0:LC%=F%
50 DIMA$(MAX%),LS%(MAX%),PT%(MAX%,1),PR%(MAX%,1)
51 :
60 PRINT"■"
70 P%=RT%:ST%=F%
80 PRINT"■COMAND--> ";:INPUT#1,X$:PRINT:REM MARIANI G.
85 PRINT"■                   □":REM 19 SPAZI
90 IFX$="**I"THENGOSUB150:GOTO70
100 IFX$="**R"THENGOSUB300:GOTO70
110 IFX$="**D"THENGOSUB360:GOTO70
120 IFX$="**S"THENGOSUB540:GOTO70
130 IFX$="**."THENCLOSE1:END
140 PRINT"ERRORE NEL COMANDO■":GOTO70
141 :
142 :
150 REM *** IMMISSIONE ***
151 :
160 IFC%>MAX%ANDLC%=F%THENPRINT"■MEMORY FULL■":RETURN
170 PRINT"DIMMI UN NOME ";:INPUT#1,X$:PRINT
180 IFLC%>F%THENCT%=LS%(LC%):GOTO200
190 CT%=C%
200 IFP%=NIL%THEN250
210 NX%=P%:ST%=F%
220 IFX$<A$(P%)THENST%=0:P%=PT%(P%,0):GOTO200
230 IFX$>A$(P%)THENST%=1:P%=PT%(P%,1):GOTO200
240 PRINT"■IL NODO ESISTE GIA'":RETURN
```

```
250 A$(CT%)=X$:PT%(CT%,0)=NIL%:PT%(CT%,1)=NIL%
260 IFRT%=NIL%THENRT%=CT%:GOTO280
270 PT%(NX%,ST%)=CT%
280 IFLC%>F%THENLC%=LC%+F%:RETURN
290 C%=C%+1:RETURN
291 :
292 :
300 REM *** CERCA ***
301 :
310 PRINT"CHIAVE ";:INPUT#1,X$:PRINT
320 IFP%=NIL%THENPRINT"█CHIAVE INESISTENTE":RETURN
330 IFX$<A$(P%)THENP%=PT%(P%,0):GOTO320
340 IFX$>A$(P%)THENP%=PT%(P%,1):GOTO320
350 PRINT"█"P%;" ";A$(P%),PT%(P%,0),PT%(P%,1):RETURN
351 :
352 :
360 REM *** CANCELLAZIONE ***
361 :
370 PRINT"DIMMI LA CHIAVE ";:INPUT#1,X$:PRINT
380 IFP%=NIL%THENPRINT"█CHIAVE INESESTENTE":RETURN
390 IFX$<A$(P%)THENST%=0:NX%=P%:P%=PT%(P%,0):GOTO380
400 IFX$>A$(P%)THENST%=1:NX%=P%:P%=PT%(P%,1):GOTO380
410 IFST%<>F%THEN450
420 Q%=P%:IFPT%(P%,0)=NIL%THENP%=PT%(P%,1):GOTO510
430 IFPT%(P%,1)=NIL%THENP%=PT%(P%,0):GOTO510
440 GOTO470
450 IFPT%(P%,0)=NIL%THENPT%(NX%,ST%)=PT%(P%,1):Q%=P%:GOTO510
460 IFPT%(P%,1)=NIL%THENPT%(NX%,ST%)=PT%(P%,0):Q%=P%:GOTO510
470 Q%=PT%(P%,0):Q1%=P%:ST%=0
480 IFPT%(Q%,1)<>NIL%THENQ1%=Q%:Q%=PT%(Q%,1):ST%=1:GOTO480
490 A$(P%)=A$(Q%)
500 PT%(Q1%,ST%)=PT%(Q%,0)
510 REM -- DISPOSE STORAGE --
520 IFST%=F%THENRT%=P%
530 LC%=LC%+1:LS%(LC%)=Q%:RETURN
531 :
532 :
540 REM *** INORDER ***
541 :
550 FORI=0TOMAX%:PR%(I,1)=0:NEXT
560 CP%=F%
580 IFPT%(P%,0)<>NIL%ANDPR%(P%,1)=0THENGOSUB650:GOTO580
590 PRINTA$(P%)
600 IFPT%(P%,1)<>NIL%THENCP%=CP%+1:PR%(CP%,0)=P%:P%=PT%(P%,1):GOTO580
610 IFCP%=F%THENRETURN
620 P%=PR%(CP%,0):CP%=CP%-1
630 IFPR%(P%,1)ORPT%(P%,0)=NIL%THEN610
640 PR%(P%,1)=1:GOTO580
650 CP%=CP%+1:PR%(CP%,0)=P%:P%=PT%(P%,0):RETURN

READY.
```

IPER MONITOR PER LINGUAGGIO MACCHINA

Sui numeri 5 e 6 di Commodore Computer Club (settembre e ottobre 1983) pubblicavamo un programma simulatore del Monitor Tim, vale a dire un programma di utilità che, ispirandosi ai modelli PET Commodore, consentiva di lavorare agevolmente in linguaggio macchina. Al termine dell'articolo che illustrava l'uso del programma invitavamo i lettori a far qualcosa di meglio e a farcelo pervenire. Dopo quasi un anno dalla pubblicazione abbiamo visto, sull'allegato di una rivista di settore, un articolo che ci faceva ben sperare. Saremmo stati lieti di utilizzarlo e (perchè no?) di consigliarlo ai nostri stessi lettori. Purtroppo dopo un esame approfondito abbiamo constatato che non era altro che una copia più o meno conforme del nostro programma, fatta eccezione di un paio di aggiunte che chiunque avrebbe potuto realizzare con un semplice comando di Append.

Delusi per la modestia del lavoro esaminato e decisi ormai a regalare ai nostri lettori un monitor potente per lavorare in linguaggio macchina, ecco che presentiamo un buon lavoro che, per chi lo conosce, simula in puro Basic, il potentissimo programma di utility denominato ZOOM. Pubblichiamo, ovviamente, solo il listato e il modo di utilizzare il programma, evitando lunghe, e, forse, noiose spiegazioni su come è strutturato. Si tenga presente che d'ora in poi, parlando in futuro di linguaggio macchina, faremo riferimento all'Output del listato pubblicato. Per ragioni di tempo, e per la lunghezza stessa del programma, viene fornita solo la versione per il Commodore 64. E' ovvio che (nonostante il programma sia stato severamente collaudato) suggerimenti e segnalazioni di eventuali anomalie saranno graditissime, come pure versioni più sofisticate di quella pubblicata.

Se la rivista nominata vorrà, anche stavolta, «ispirarsi» ai nostri listati, ne saremo lietissimi. Però, diamine!, mettecela tutta. Programmare non è difficile: lo dimostra la giovanissima età dell'autore del listato (16 anni) che in soli due giorni di lavoro ha realizzato ciò che vedete.

I comandi dell'Iper Monitor

• Primo comando: "M". Esempio:

M 0800 0900

Visualizza la zona di memoria indicata in esadecimale. Come separatore tra i due indirizzi si può usare il carattere spazio oppure il segno meno (-). Per fermare temporaneamente la visualizzazione premere il tasto Shift (Shift - lock se per lungo tempo). Per fermarla definitivamente basta un tasto qualsiasi.

• Secondo comando: "↑". Esempio:

↑ 033A A9 00 8D 00 04 60

Consente di scrivere nella memoria i valori esadecimali digitati. In questo caso a partire da 033A inserirà l'uno dopo l'altro A9, 00 ecc. Dopo aver digitato come descritto è sufficiente premere il tasto Return per la conferma (come nei veri monitor). Subito il programma scriverà l'indirizzo successivo e si predisporrà per una seconda scrittura. Se non si desidera scrivere altro, battete un puntino (.) e il tasto Return.

• Terzo comando: "I". Esempio.

I 033A 034F

Visualizza il contenuto della zona di memoria indicata espresso non come cifra esadecimale ma come carattere ASCII. Se un particolare valore non è contemplato nel codice ASCII, viene visualizzato, al suo posto, un puntino. Attenzione, quindi, ad interpretare correttamente ciò che appare. In questi casi è meglio esaminare la stessa zona con il comando "M". Per la separazione di indirizzi, rallentamento e interruzione vedi il comando M.

• Quarto comando: "@" (chiocciolina). Esempio:

@ 033A 'CARATTERI ASCII

Consente di trasferire in memoria, a partire dall'indirizzo esadecimale digitato, opportunatamente tradotti, i valori corrispondenti alle lettere C, A, R, eccetera. Si badi bene di digitare: chiocciolina, spazio, l'indirizzo, uno spazio, l'apostrofo ed in seguito i caratteri desiderati. Anche in questo caso alla fine dei caratteri digitati (max =26) battere return. Verranno trascritti solo i caratteri digitati effettivamente. Se ad esempio si digita:

@ 033A 'PAROLA

a partire da 033A verranno trascritti, opportunatamente tradotti, i sei caratteri che costituiscono la parola digitata; gli altri resteranno immutati.

Dopo aver operato, il programma visualizzerà l'indirizzo successivo all'ulti-

mo utilizzato e attenderà che si digitino altri caratteri. Per finire è necessario digitare il carattere "π" (pi-greco).
• Quinto comando: "R"

Visualizza il contenuto dei vari registri:
Ac (accumulatore), XR (registro X), YR (registro Y), SP (Stack Pointer).
● Sesto comando ">". Esempio:
> A9 C0 FF A5

Consente di modificare il contenuto dei registri.
● Settimo comando: "F". Esempio:
F 033A 033F 00

Riproduce in tutte le locazioni di memoria, a partire dal primo fino al secondo indirizzo, il valore 00.
● Ottavo comando: "C". Esempio:
C 033A 03FF 0400

Confronta i contenuti delle locazioni di memoria a partire da 033A fino a 03FF con i corrispondenti byte allocati a partire da 0400. Nel caso i due blocchi siano eguali non viene visualizzato nulla. Se invece vi sono discordanze appaiono i valori esadecimali degli indirizzi in cui è stata riscontrata la differenza. Per interrompere la visualizzazione premere un tasto qualsiasi.
● Nono comando: "T". Esempio:
T 0334 033B 033A

Trasferisce i contenuti delle locazioni, da 0334 fino a 033B, a partire da 033A. Attenzione a questo, come ad altri comandi, a non invadere l'area in cui lavora il programma stesso!
● Decimo comando: "H". Esempio:
H 0400 0800 55

Ricerca all'interno dell'area 0400-0800, il byte che contiene il valore esadecimale 55 e ne visualizza l'indirizzo. Un tasto qualsiasi, come al solito, interrompe la ricerca.
● Undicesimo comando: "G". Esempio:
G 033A

Fa eseguire la routine in linguaggio macchina a partire da 033A (equivale a SYS). Se questa routine termina con 60 (RTS) viene restituito il controllo al programma.
● Dodicesimo comando: "$". Esempio:
$ 56FE

Traduce il numero esadecimale nel corrispondente decimale. Il numero deve essere sempre di quattro cifre; pertanto non digitare A0, ma 00A0.
● Tredicesimo comando: "≠". Esempio:
≠ 12765

Inverso di "$". Il valore deve essere sempre di cinque cifre; non 160, ma 00160.
● Quattordicesimo comando: "P"

Attiva la stampante. Da questo momento la visualizzazione verrà trasferita alla stampante e non più al video.
● Quindicesimo comando: "O"

Disattiva la stampante e abilita nuovamente il video.
3● Sedicesimo comando: "X"

Restituisce il controllo al Basic interrompendo il programma.
● Diciassettesimo comando. "L". Esempio:
L 'NOME' 08

Carica un programma da nastro (01) o disco (08) a partire dalla locazione in cui era al momento della registrazione. Se viene riscontrato un errore appare il messaggio ERROR.
● Diciottesimo comando: "S". Esempio:
S 'ROUTINE' 08 033A 03FB

Registra col nome ROUTINE su disco (08) il programma in linguaggio macchina allocato da 033A fino a 03FB.
● Diciannovesimo comando: "V"

Verifica il programma registrato.
● Ventesimo comando: "*"

Consente di conoscere l'indirizzo di partenza di un programma registrato su nastro.

● Comandi per il disco. Sono tutti preceduti dal carattere "←".

←$

Visualizza la directory del disco.

←NO/NOME.XX

Formatta il disco conferendogli il nome NOME e l'identificatore XX.

←I0

Inizializza il disco.

In altre parole i comandi per il disco si possono usare con la forma appena vista sostituendo agli apici (') la barra (/), e alle virgole (,) il punto (.). Il solo carattere di freccia a sinistra visualizza lo stato del disco.

← ≠NOME.

Consente di conoscere l'indirizzo di partenza del programma residente su disco chiamato NOME.
● Ultimo comando: RUN
Consente, se avete combinato qualche pasticcio, di far ripartire il programma.

Avvertenze

Dopo aver digitato con pazienza il programma, *registratelo prima di utilizzarlo.* In seguito iniziate a verificare la funzionalità dei comandi utilizzando solo PEEK come "M", "I", "C" eccetera. Il programma Basic occupa circa 8 KRAM; la sua esatta occupazione la potete individuare da soli.

Iper Monitor consente addirittura di alterare il vettore relativo all'interrupt dato che, prima di trascrivere i dati digitati, esegue una routine in linguaggio macchina che maschera l'interrupt. Il programma utilizza inoltre tutti i byte da 680 a 727 ed i byte 251 e 252. Attenzione a non alterarli nei vostri programmi. Il programma utilizza, in alcuni casi, il buffer della tastiera.

Non crediamo di aver dimenticato nulla salvo il fatto che, benchè sia superfluo, vi consigliamo di non alterare il contenuto riservato al programma Basic che è allocato da 0800 fino a... No, non vogliamo dirvelo: scopritelo con l'Iper Monitor in uno dei vostri viaggi nei meandri della memoria.

Giancarlo Mariani (da un suggerimento di A. de Simone)

```
100 REM *****    IPER MONITOR PER COMMODORE 64    *****
110 REM *****      DI   GIANCARALO MARIANI          *****
120 REM ***** VIALE BRIANZA 72 20036 MEDA (MI) *****
130 REM *****            TEL. 0362/72565            *****
131 :
132 :
140 POKE53280,1:POKE53281,1:GOTO2870
141 REM NON RENUMERARE LA LINEA 150
142 REM PER NESSUN MOTIVO (CFR. LINEA 2870)
150 POKE198,0
160 PRINT"██":POKE56,40:POKE52,40:CLR:OPEN1,0
170 PRINT"MONITOR 64"
180 PRINT"BY G.MARIANI  TEL.0362/72565█"
190 L=-1:GOSUB1430
200 C$="":INPUT#1,C$:PRINT:A9=3
210 IFC$="RUN"THENRUN
220 A$=LEFT$(C$,1):L=LEN(C$)-2
230 IFA$="F"THENGOSUB450:GOTO200
240 IFA$="X"THENCLOSE1:CLR:END
250 IFA$="$"THENGOSUB530:GOTO200
260 IFA$="#"THENGOSUB580:GOTO200
270 IFA$="G"THENGOSUB640:GOTO200
280 IFA$="C"THENGOSUB680:GOTO200
290 IFA$="M"THENGOSUB820:GOTO200
300 IFA$="P"ANDL=-1THENP=1:GOTO200
310 IFA$="O"ANDL=-1THENP=0:GOTO200
320 IFA$="I"THENGOSUB1000:GOTO200
330 IFA$="T"THENGOSUB1200:GOTO200
340 IFA$="↑"THENGOSUB1300:GOTO200
350 IFA$="R"THENGOSUB1430:GOTO200
360 IFA$=">"THENGOSUB1490:GOTO200
370 IFA$="@"THENGOSUB1550:GOTO200
380 IFA$="*"THENGOSUB1690:GOTO200
390 IFA$="←"THENGOSUB1770:GOTO200
400 IFA$="H"THENGOSUB2240:GOTO200
410 IFA$="L"ORA$="V"THENGOSUB2390:GOTO200
420 IFA$="S"THENGOSUB2610:GOTO200
```

```
430 IFC$<>""THENGOSUB2880
440 GOTO200
450 REM ********** F (RIEMPIE) **********
460 IFL<>12THENGOSUB2880:RETURN
470 X$=MID$(C$,7,1):IFX$<>" "ANDX$<>"-"THENGOSUB2880:RETURN
480 ES$="00"+RIGHT$(C$,2):GOSUB2950:A3=X:IFV=1THENV=0:RETURN
490 ES$=MID$(C$,8,4):GOSUB2950:A2=X:IFV=1THENV=0:RETURN
500 ES$=MID$(C$,3,4):GOSUB2950:A1=X:IFV=1THENV=0:RETURN
510 IFA1>A2THENGOSUB2880:RETURN
520 FORK=A1TOA2:POKEK,A3:NEXT:RETURN
530 REM ********** ESAD/DEC **********
540 V=0:C$=RIGHT$(C$,4)
550 IFL<>4THENGOSUB2880:RETURN
560 ES$=C$:GOSUB2950:IFV=1THENV=0:RETURN
570 PRINT"#";X:RETURN
580 REM ********** DEC/ESAD **********
590 IFL<=0THENGOSUB2880:RETURN
600 C$=RIGHT$(C$,L)
610 X=VAL(C$):IFX>65535THENGOSUB2880:RETURN
620 GOSUB2900
630 PRINT"$ ";ES$:RETURN
640 REM ********** G (VAI) **********
650 IFL<>4THENGOSUB2880:RETURN
660 ES$=RIGHT$(C$,4):GOSUB2950:IFV=1THENV=0:RETURN
670 SYSX:RETURN
680 REM ********** C (CONFRONTA) **********
690 POKE198,0
700 IFL<>14THENGOSUB2880:RETURN
710 C$=RIGHT$(C$,L)
720 A$=MID$(C$,5,1):IFA$<>" "ANDA$<>"-"THENGOSUB2880:RETURN
730 A$=MID$(C$,10,1):IFA$<>" "ANDA$<>"-"THENGOSUB2880:RETURN
740 ES$=LEFT$(C$,4):GOSUB2950:A1=X:IFV=1THENV=0:RETURN
750 ES$=MID$(C$,6,4):GOSUB2950:A2=X:IFV=1THENV=0:RETURN
760 ES$=RIGHT$(C$,4):GOSUB2950:A3=X:IFV=1THENV=0:RETURN
770 IFA1>A2THENGOSUB2880:RETURN
780 FORKK=0TO(A2-A1)-1
790 IFPEEK(A1+KK)<>PEEK(A3+KK)THENX=A1+KK:GOSUB2900:PRINTES$;" ";
800 IFPEEK(197)<>64THENPOKE198,0:PRINT:RETURN
810 NEXT:PRINT:RETURN
820 REM ********** M (VISUALIZZA IN HEX) **********
830 IFP=1THENOPEN4,4:CMD4
840 IFL<>9THENGOSUB2880:RETURN
850 C$=RIGHT$(C$,9)
860 A$=MID$(C$,5,1):IFA$<>" "ANDA$<>"-"THENGOSUB2880:RETURN
870 ES$=LEFT$(C$,4):GOSUB2950:A1=X:IFV=1THENV=0:RETURN
880 ES$=RIGHT$(C$,4):GOSUB2950:A2=X:IFV=1THENV=0:RETURN
890 IFA1>A2THENGOSUB2880:RETURN
900 PRINT
910 FORKK=A1TOA2STEP8
920 X=KK:GOSUB2900
930 PRINT"↑ ";ES$;" ";:FORKX=KKTOKK+7:X=PEEK(KX):A9=1:GOSUB2900
940 PRINTES$;" ";:NEXTKX:PRINT
```

```
950  IFPEEK(653)<>0THEN950
960  IFPEEK(197)<>64THEN980
970  NEXTKK
980  PRINT:IFP=1THENPRINT#4:CLOSE4
990  POKE198,0:RETURN
1000 REM ********** I (VISUALIZZA IN ASCII) **********
1010 IFP=1THENOPEN4,4:CMD4
1020 IFL<>9THENGOSUB2880:RETURN
1030 C$=RIGHT$(C$,9)
1040 A$=MID$(C$,5,1):IFA$<>" "ANDA$<>"-"THENGOSUB2880:RETURN
1050 ES$=LEFT$(C$,4):GOSUB2950:A1=X:IFV=1THENV=0:RETURN
1060 ES$=RIGHT$(C$,4):GOSUB2950:A2=X:IFV=1THENV=0:RETURN
1070 IFA1>A2THENGOSUB2880:RETURN
1080 PRINT
1090 FORKK=A1TOA2STEP26
1100 X=KK:GOSUB2900
1110 PRINT"@ ";ES$" '";:FORKX=KKTOKK+25:X=PEEK(KX)
1120 X$=".":IFX>=32ANDX<=127THENX$=CHR$(X)
1130 IFX>=160THENX$=CHR$(X)
1140 PRINTX$;:NEXTKX:PRINT
1150 IFPEEK(653)<>0THEN1150
1160 IFPEEK(197)<>64THEN1180
1170 NEXTKK
1180 PRINT:IFP=1THENPRINT#4:CLOSE4
1190 POKE198,0:RETURN
1200 REM ********** T (TRASFERISCE) **********
1210 IFL<>14THENGOSUB2880:RETURN
1220 C$=RIGHT$(C$,14)
1230 A$=MID$(C$,5,1):IFA$<>" "ANDA$<>"-"THENGOSUB2880:RETURN
1240 A$=MID$(C$,10,1):IFA$<>" "ANDA$<>"-"THENGOSUB2880:RETURN
1250 ES$=LEFT$(C$,4):GOSUB2950:A1=X:IFV=1THENV=0:RETURN
1260 ES$=MID$(C$,6,4):GOSUB2950:A2=X:IFV=1THENV=0:RETURN
1270 ES$=RIGHT$(C$,4):GOSUB2950:A3=X:IFV=1THENV=0:RETURN
1280 IFA1>A2THENGOSUB2880:RETURN
1290 FORK=0TO(A2-A1)-1:POKEA3+K,PEEK(A1+K):NEXT:RETURN
1300 REM ********** ↑ (SCRIVE IN HEX) **********
1310 GOTO1340
1320 C$="":INPUT#1,C$:PRINT:L=LEN(C$)-2
1330 IFLEFT$(C$,1)<>"↑"THENC$="↑ "+ES$+" "+C$:L=L+7
1340 IFMID$(C$,8,1)="."THENRETURN
1350 IFL<6ORL>28THENGOSUB2880:RETURN
1360 ES$=MID$(C$,3,4):GOSUB2950:A1=X:IFV=1THENV=0:RETURN
1370 C$=RIGHT$(C$,LEN(C$)-7)+" ":L=LEN(C$)
1380 A2=0:POKE56334,0:FORK=1TOLSTEP3
1390 ES$=MID$(C$,K,2):A9=1:GOSUB2950
1400 IFV=1THENV=0:POKE56334,1:RETURN
1410 POKEA1+A2,X:A2=A2+1:NEXT:POKE56334,1:A1=A1+A2
1420 X=A1:GOSUB2900:PRINT"↑ ";ES$" ";:GOTO1320
1430 REM ********** R (VISUALIZZA REGISTRI) **********
1440 IFL>=0THENGOSUB2880:RETURN
1450 PRINT"  AC XR YR SP":PRINT"> ";
1460 FORK=780TO783
```

COMMODORE SCHOOL

```
1470 X=PEEK(K):A9=1:GOSUB2900:PRINTES$" ";:NEXT
1480 PRINT:RETURN
1490 REM ********** > (MODIFICA REGISTRI) **********
1500 IFL<>11THENGOSUB2880:RETURN
1510 C$=RIGHT$(C$,11)+" "
1520 A2=0:FORK=1TO12STEP3:ES$="00"+MID$(C$,K,2)
1530 GOSUB2950:IFV=1THENV=0:RETURN
1540 POKE780+A2,X:A2=A2+1:NEXT:RETURN
1550 REM ********** @ (SCRIVE IN ASCII) **********
1560 GOTO1590
1570 C$="":INPUT#1,C$:PRINT
1580 IFLEFT$(C$,2)<>"@ "THENC$="@ "+ES$+" '"+C$
1590 L=LEN(C$):IFL<9THENGOSUB2880:RETURN
1600 ES$=MID$(C$,3,4):GOSUB2950:A1=X:IFV=1THENV=0:RETURN
1610 ES$=MID$(C$,3,4):GOSUB2950:A1=X:IFV=1THENV=0:RETURN
1620 C$=RIGHT$(C$,LEN(C$)-8):L=LEN(C$)
1630 IFL<1ORL>26THENGOSUB2880:RETURN
1640 IFLEFT$(C$,1)="←"THENRETURN
1650 POKE56334,0:AA=1:FORKK=A1TOA1+L-1
1660 POKEKK,ASC(MID$(C$,AA,1)):AA=AA+1:NEXT
1670 POKE56334,1
1680 A1=A1+L:X=A1:GOSUB2900:PRINT"@ "ES$" '";:GOTO1570
1690 REM ********** (INIZIO & FINE DA CASS) **********
1700 IFL>15THENGOSUB2880:RETURN
1710 C$=RIGHT$(C$,LEN(C$)-1)
1720 OPEN10,1,0,C$:CLOSE10
1730 X=PEEK(830)*256+PEEK(829)
1740 GOSUB2900:PRINT"█"ES$"-";
1750 X=PEEK(832)*256+PEEK(831)
1760 GOSUB2900:PRINTES$"█":RETURN
1770 REM ********** ← (COMANDI DISCO) **********
1780 C$=RIGHT$(C$,LEN(C$)-1):A$=LEFT$(C$,1)
1790 IFA$="$"THENGOSUB2030:RETURN
1800 IFA$="#"THENGOSUB1920:RETURN
1810 REM ********** COMANDI **********
1820 FORK=1TOLEN(C$)
1830 IFMID$(C$,K,1)="/"THENC$=LEFT$(C$,K-1)+":"+RIGHT$(C$,L-K+1)
1840 IFMID$(C$,K,1)="."THENC$=LEFT$(C$,K-1)+","+RIGHT$(C$,L-K+1)
1850 NEXT
1860 OPEN15,8,15,C$
1870 GOSUB1890
1880 RETURN
1890 REM ********** ERRORE **********
1900 INPUT#15,A1$,A2$,A3$,A4$
1910 PRINT"██"A1$","A2$","A3$","A4$"█":CLOSE15:RETURN
1920 REM ********** INIZIO PROG DA DISCO **********
1930 C$=RIGHT$(C$,LEN(C$)-1):IFLEN(C$)=0THENGOSUB2880:RETURN
1940 OPEN10,8,3,C$
1950 GET#10,A1$:GET#10,A2$:CLOSE10
1960 X$=A1$:GOSUB2010:A1=A
1970 X$=A2$:GOSUB2010:A2=A*256
1980 X=A1+A2:GOSUB2900:PRINT"█"ES$
```

```
1990 OPEN15,8,15:GOSUB1900
2000 RETURN
2010 IFX$=""THENA=0:RETURN
2020 A=ASC(X$):RETURN
2030 REM ********** DIRECTORY **********
2040 POKE198,0
2050 PRINT
2060 OPEN10,8,0,"$0"
2070 GET#10,A$,B$
2080 GET#10,A$,B$:GET#10,A$,B$:C=0
2090 IFA$<>""THENC=ASC(A$)
2100 IFB$<>""THENC=C+ASC(B$)*256
2110 PRINTMID$(STR$(C),2);TAB(4);
2120 GET#10,B$:IFST<>0THEN2220
2130 IFB$<>CHR$(34)THEN2120
2140 GET#10,B$:IFB$<>CHR$(34)THENPRINTB$;:GOTO2140
2150 GET#10,B$:IFB$=CHR$(32)THEN2150
2160 PRINTTAB(18);:C$=""
2170 C$=C$+B$:GET#10,B$:IFB$<>""THEN2170
2180 PRINT"  "LEFT$(C$,3)
2190 IFPEEK(653)<>0THEN2190
2200 IFPEEK(197)<>64THENPOKE198,0:GOTO2230
2210 IFST=0THEN2080
2220 PRINT" BLOCKS FREE■"
2230 CLOSE10:RETURN
2240 REM ********** H (RICERCA) **********
2250 IFL<>12THENGOSUB2880:RETURN
2260 POKE198,0
2270 C$=RIGHT$(C$,L)
2280 X$=MID$(C$,10,1):IFX$<>" "ANDX$<>"-"THENGOSUB2880:RETURN
2290 X$=MID$(C$,5,1):IFX$<>" "ANDX$<>"-"THENGOSUB2880:RETURN
2300 ES$=LEFT$(C$,4):GOSUB2950:A1=X:IFV=1THENRETURN
2310 ES$=MID$(C$,6,4):GOSUB2950:A2=X:IFV=1THENRETURN
2320 ES$="00"+RIGHT$(C$,2):GOSUB2950:A3=X:IFV=1THENRETURN
2330 IFA1>A2THENGOSUB2880:RETURN
2340 PRINT:FORKK=A1TOA2
2350 IFPEEK(KK)=A3THENX=KK:GOSUB2900:PRINTES$;" ";
2360 IFPEEK(197)<>64THEN2380
2370 NEXT
2380 POKE198,0:PRINT:RETURN
2390 REM ********** L & V (LOAD & VERIFY) **********
2400 AA$=A$
2410 L=L+2:IFL<6THENGOSUB2880:RETURN
2420 DATA 169,15,162,0,160,1,32,186,255,169
2430 DATA 0,162,198,160,2,32,189,255
2440 DATA 169,0,162,0,160,0,32,213,255,96
2450 C$=RIGHT$(C$,LEN(C$)-1)
2460 A2=VAL(RIGHT$(C$,2)):C$=LEFT$(C$,LEN(C$)-3)
2470 IFA2<>1ANDA2<>8THENGOSUB2880:RETURN
2480 A$=LEFT$(C$,1):B$=RIGHT$(C$,1)
2490 IFA$<>"'"ORB$<>"'"THENGOSUB2880:RETURN
2500 C$=LEFT$(C$,LEN(C$)-1):C$=RIGHT$(C$,LEN(C$)-1)
```

```
2510 IFA2=8ANDC$=""THENGOSUB2880:RETURN
2520 IFLEN(C$)>16THENGOSUB2880:RETURN
2530 RESTORE:FORT=0TO27:READA:POKE680+T,A:NEXT
2540 IFC$=""THEN2560
2550 FORK=1TOLEN(C$):POKE709+K,ASC(MID$(C$,K,1)):NEXT
2560 POKE683,A2:POKE690,LEN(C$):POKE699,0:IFAA$="V"THENPOKE699,1
2570 SYS680
2580 IFA2=1ANDST<>0THENPRINT"   ERROR"
2590 IFA2=8ANDST=80THENPRINT"   ERROR"
2600 PRINT:RETURN
2610 REM ********** S (SAVE) **********
2620 L=L+2:IFL<16THENGOSUB2880:RETURN
2630 DATA 169,15,162,0,160,1,32,186,255
2640 DATA 169,0,162,198,160,2,32,189,255,169,251
2650 DATA 162,0,160,0,32,216,255,96
2660 C$=RIGHT$(C$,LEN(C$)-1)
2670 ES$=RIGHT$(C$,4):C$=LEFT$(C$,LEN(C$)-5):GOSUB2950:A2=X
2680 ES$=RIGHT$(C$,4):C$=LEFT$(C$,LEN(C$)-5):GOSUB2950:A1=X
2690 IFA1>=A2THENGOSUB2880:RETURN
2700 A3=VAL(RIGHT$(C$,2)):IFA3<>1ANDA3<>8THENGOSUB2880:RETURN
2710 C$=LEFT$(C$,LEN(C$)-3)
2720 A$=LEFT$(C$,1):B$=RIGHT$(C$,1)
2730 IFA$<>"'"ORB$<>"'"THENGOSUB2880:RETURN
2740 C$=LEFT$(C$,LEN(C$)-1):C$=RIGHT$(C$,LEN(C$)-1)
2750 IFLEN(C$)>16THENGOSUB2880:RETURN
2760 A(2)=INT(A1/256):A(1)=A1-A(2)*256
2770 A(4)=INT(A2/256):A(3)=A2-A(4)*256
2780 RESTORE:FORT=0TO27:READA:NEXT
2790 FORT=0TO27:READA:POKE680+T,A:NEXT
2800 IFC$=""THEN2820
2810 FORK=1TOLEN(C$):POKE709+K,ASC(MID$(C$,K,1)):NEXT
2820 POKE683,A3:POKE690,LEN(C$)
2830 POKE701,A(3):POKE703,A(4):POKE251,A(1)
2840 POKE252,A(2):SYS680:PRINT
2850 RETURN
2860 END
2870 PRINT"CLR:GOTO150 ";:POKE631,13:POKE198,1:END
2880 REM ********** ERRORE **********
2890 PRINT"□";TAB(37);"?":RETURN
2900 REM ********** DEC-ESA **********
2910 ES$="":FORK9=0TOA9:X1=INT(X/16↑(A9-K9))
2920 X=X-X1*16↑(A9-K9):AK$=STR$(X1)
2930 E1$=RIGHT$(AK$,LEN(AK$)-1):IFX1>9THENE1$=CHR$(X1+55)
2940 ES$=ES$+E1$:NEXT:A9=3:RETURN
2950 REM ********** ESA-DEC **********
2960 X=0:FORK9=0TOA9:X1$=MID$(ES$,K9+1,1):U=VAL(X1$)
2970 IFU=0ANDX1$<>"0"THEN2990
2980 X=X+U*16↑(A9-K9):NEXT:A9=3:RETURN
2990 U=ASC(X1$)-55:IFU>15ORU<-7THENV=1:GOSUB2880:RETURN
3000 X=X+U*16↑(A9-K9):NEXT:A9=3:RETURN

READY.
```

COMMODORE SCHOOL

DISASSEMBLER

Quest'utile programma, numerato da 4000 in poi in modo da consentire l'"APPEND" al programma Iper Monitor, permette di esplorare la memoria del vostro computer. Le istruzioni per il funzionamento sono riportate da 4040 a 4180.

Se, utilizzando il programma, appare l'errore della riga 4232, vuol dire che avete digitato erroneamente alcuni dei valori 1,2,3 riportati nei DATA. L'errore di riga 4233 indica, invece, una errata trascrizione di asterischi (*).

La figura riportata rappresenta l'output della routine CHARGET utilizzata nel programma ESPANSIONE DEI COMANDI BASIC.

IND.N	ESA	DATI	MNEMON.	
115	0073	E6 7A	INCZ	122
117	0075	D0 02	BNE	2
119	0077	E6 7B	INCZ	123
121	0079	AD 26 0F	LDA	3944
124	007C	C9 3A	CMPIM	58
126	007E	B0 0A	BCS	10
128	0080	C9 20	CMPIM	32
130	0082	F0 EF	BEQ	239
132	0084	38	SEC	
133	0085	E9 30	SBCIM	48
135	0087	38	SEC	
136	0088	E9 D0	SBCIM	208
138	008A	60	RTS	

```
4000 REM ** DISASSEMBLER 6502-6510 **
4010 REM **                        **
4020 REM ** ALESSANDRO  DE SIMONE  **
4030 :
4040 PRINT"      DIGITA IN DECIMALE L'INDIRIZZO DI
4050 PRINT"PARTENZA DEL PROGRAMMA DA DISASSEMBLARE.
4060 PRINT:PRINT"DOPO CHE SULLO SCHERMO (O SU STAMPANTE)
4070 PRINT"SONO APPARSE 23 RIGHE, E' NECESSARIO
4080 PRINT"DIGITARE RETURN PER CONTINUARE OPPURE
4090 PRINT"HOME PER TERMINARE O SPACE PER OTTENERE
4100 PRINT"LA SOLA ISTRUZIONE SUCCESSIVA.":PRINT
4110 PRINT"  CON NULL SONO INDICATI I DATI CHE"
4120 PRINT"NON SONO RICONOSCIUTI COME ISTRUZIONI.":PRINT:PRINT
4130 PRINT"  UTILIZZANDO LA STAMPANTE APPAIONO
4140 PRINT"SIA SU VIDEO SIA SU STAMPANTE LE
4150 PRINT"TRADUZIONI ESADECIMALE- DECIMALE E
4160 PRINT"VICEVERSA
4170 PRINT"PER INTERROMPERE PREMERE SHIFT"
4180 PRINT"PER FINIRE BATTERE UN INDIRIZZO NEGATIVO"
4190 DIMMN$(256),BY%(256),CO$(16)
4200 FORE=0TO255:READMN$(E)
4210 IFMN$(E)="*"THENBY%(E)=0:MN$(E)="NULL":GOTO4230
4220 READBY%(E)
4230 NEXT
4231 Y=0:X=0:FORI=0TO255:X=X+BY%(I):IFBY%(I)=0THENY=Y+1
```

```
4232 NEXT:IFX<>321THENPRINT"1/2/3 ERRATI NEI DATA":END
4233 IFY<>105THENPRINT"ERRORI IN QUANTITA' DI * DATA":END
4240 FORE=0TO15
4250 READCO$(E)
4260 NEXT
4270 PRINT:PRINT"                    PREMI UN TASTO"
4280 GETX$:IFX$=""THEN4280
4290 GOTO4360
4300 SX=INT(DC/16)
4310 UN=DC-(SX*16)
4320 SX$=CO$(SX)
4330 UN$=CO$(UN)
4340 HX$=SX$+UN$
4350 RETURN
4360 PRINT:PRINT"CON STAMPA ? SI O NO";
4370 GETX$:IFX$=""THEN4370
4380 IFX$<>"S"THEN4410
4390 OPEN4,4
4400 F1$="IND.N ESA    DATI       MNEMON."
4410 PRINT"";
4420 INPUT"INDIRIZZO DI PARTENZA";AD:I=0
4430 IF AD<0 THEN CLOSE4:END
4440 IFX$="S"THENIFI=0THENPRINT#4,F1$:PRINT#4
4450 IFI=23GOTO4740
4460 I=I+1
4470 IB=PEEK(AD)
4480 IF PEEK(653) THEN 4420
4490 ONBY%(IB)GOTO4500,4550,4630
4500 DC=IB:GOSUB4300:GOSUB5100
4510 PRINTAD;AD$;TAB(12);HX$;TAB(21);MN$(IB)
4520 IFX$="S"THENPRINT#4,AD"  "AD$,HX$"          ",MN$(IB)
4530 AD=AD+1
4540 GOTO4730
4550 DC=IB:GOSUB4300
4560 B1$=HX$
4570 DC=PEEK(AD+1):GOSUB4300
4580 B2$=HX$
4590 GOSUB5100:P=DC
4600 PRINTAD;AD$TAB(12);B1$;" ";B2$;TAB(21);MN$(IB);TAB(27);P
4610 IFX$="S"THENPRINT#4,AD"  "AD$,B1$"  "B2$"    ",MN$(IB),P
4620 AD=AD+2:GOTO4730
4630 DC=IB:GOSUB4300:B1$=HX$:DC=PEEK(AD+1):GOSUB4300
4640 B2$=HX$
4650 DC=PEEK(AD+2):GOSUB4300
4660 B3$=HX$
4670 OP=PEEK(AD+1)+(PEEK(AD+2)*256)
4680 GOSUB5100
4690 PRINTAD;AD$;TAB(12);B1$;" ";B2$;" ";
4700 PRINTB3$;TAB(21);MN$(IB);TAB(27);OP
4710 IFX$="S"THENPRINT#4,AD"  "AD$,B1$"  "B2$"  "B3$,MN$(IB),OP
4720 AD=AD+3
```

```
4730 GOTO4440
4740 GETA$:IFA$=""THEN4740
4750 IFA$=CHR$(19)THENCLOSE4:GOTO4410
4760 IFA$=" "THENI=22:GOTO4440
4770 IFA$<>CHR$(13)THEN4740
4780 I=0
4790 GOTO4440
4800 DATA BRK,1,ORAIX,2,*,*,*,ORAZ,2,ASL,2,*,PHP,1
4810 DATA ORAIM,2,ASLA,1,*,*,ORA,3,ASL,3,*,BPL,2,ORAIY,2
4820 DATA *,*,*,ORAZX,2,ASLZX,2,*,CLC,1,ORAY,3
4830 DATA *,*,*,ORAX,3,ASLX,3,*,JSR,3,ANDIX,2,*
4840 DATA *,BITZ,2,ANDZ,2,ROLZ,2,*,PLP,1,ANDIM,2,ROLA,1,*
4850 DATA BIT,3,AND,3,ROL,3,*,BMI,2,ANDIY,2,*,*,*
4860 DATA ANDZX,2,ROLZX,2,*,SEC,1,ANDY,3,*,*,*,ANDX,3
4870 DATA ROLX,3,*,RTI,1,EORIX,2,*,*,*,EORZ,2,LSRZ,2
4880 DATA *,PHA,1,EORIM,2,LSRA,1,*,JMP,3,EOR,3,LSR,3,*
4890 DATA BVC,2,EORIY,2,*,*,*,EORZX,2,LSRZX,2,*
4900 DATA CLI,1,EORY,3,*,*,*,EORX,3,LSRX,3,*,RTS,1
4910 DATA ADCIX,2,*,*,*,ADCZ,2,RORZ,2,*,PLA,1,ADCIM,2
4920 DATA RORA,1,*,JMPI,3,ADC,3,ROR,3,*,BVS,2,ADCIY,2,*
4930 DATA *,*,ADCZX,2,RORZX,2,*,SEI,1,ADCY,3,*,*
4940 DATA *,ADCX,3,RORX,3,*,*,STAIX,2,*,*,STYZ,2
4950 DATA STAZ,2,STX,2,*,DEY,1,*,TXA,1,*,STY,3,STA,3
4960 DATA STX,3,*,BCC,2,STAIY,2,*,*,STYZX,2,STAZX,2,STXZY,2
4970 DATA *,TYA,1,STAY,3,TXS,1,*,*,STAX,3,*,*
4980 DATA LDYIM,2,LDAIX,2,LDXIM,2,*,LDYZ,2,LDAZ,2,LDXZ,2,*
4990 DATA TAY,1,LDAIM,2,TAX,1,*,LDY,3,LDA,3,LDX,3,*,BCS,2
5000 DATA LDAIY,2,*,*,LDYZX,2,LDAZX,2,LDXZY,2,*,CLV,1
5010 DATA LDAY,3,TSX,1,*,LDYX,3,LDAX,3,LDXY,3,*,CPYIM,2,CMPIX,2
5020 DATA *,*,CPYZ,2,CMPZ,2,DECZ,2,*,INY,1,CMPIM,2,DEX,1
5030 DATA *,CPY,3,CMP,3,DEC,3,*,BNE,2,CMPIY,2,*,*
5040 DATA *,CMPZX,2,DECZX,2,*,CLD,1,CMPY,3,*,*,*
5050 DATA CMPX,3,DECX,3,*,CPXIM,2,SBCIX,2,*,*,CPXZ,2,SBCZ,2
5060 DATA INCZ,2,*,INX,1,SBCIM,2,NOP,1,*,CPX,3,SBC,3,INC,3
5070 DATA *,BEQ,2,SBCIY,2,*,*,*,SBCZX,2,INCZX,2,*,SED,1
5080 DATA SBCY,3,*,*,*,SBCX,3,INCX,3,*
5090 DATA 0,1,2,3,4,5,6,7,8,9,A,B,C,D,E,F
5100 A=AD:S3=INT(AD/4096)
5110 A=A-S3*4096
5120 S2=INT(A/256)
5130 A=A-S2*256
5140 S=INT(A/16)
5150 U=AD-(S3*4096+S2*256+S*16)
5160 S3$=CO$(S3)
5170 S2$=CO$(S2)
5180 S$=CO$(S)
5190 U$=CO$(U)
5200 AD$=S3$+S2$+S$+U$
5210 RETURN

READY.
```

ESPANSIONE DEI COMANDI BASIC

Un modo di inserire comandi personalizzati nel vostro computer.

Questo articolo, e il relativo programma, si rivolgono principalmente ai lettori che già conoscono approfonditamente il Commodore 64 e sanno cavarsela con la programmazione in Assembler.

Per tutti gli altri interessati al solo uso di nuovi comandi (chissà che anche voi non vi prendiate una cotta per l'Assembler prima o poi), vi è anche il programma in Basic che vi permette di aggiungere l'utile comando ←C A,B dove 'A' e 'B' sono rispettivamente il colore del bordo e dello sfondo. Basta caricare il programma e dare RUN per avere a disposizione il nuovo comando.

Per i lettori più esperti non dovrebbe essere difficile capire il funzionamento del programma, ma in ogni caso ecco la descrizione dello stesso. Fortunatamente i progettisti del sistema operativo del C 64 hanno previsto che sarebbe stato utile poter espandere in qualche modo i comandi del Basic, e al fine di permettere ciò hanno situato una importante routine in RAM in modo da poter essere convenientemente modificata all'occorrenza.

Tale routine si occupa di cercare il prossimo carattere da interpretare ignorando eventuali spazi bianchi. Il programma pubblicato modifica questa routine che è situata in pagina zero a partire dalla locazione 115. Se esaminate attentamente il listato dell'Assemblatore (tenete presente che si tratta dell'Assemblatore pubblicato sul numero 7 di questa stessa rivista, per cui controllate la particolare sintassi usata), vedrete che il tutto è molto semplice.

In pratica si sostituiscono i primi 3 bytes della routine originale con una istruzione di salto alle nostre routines che provvedono, naturalmente, a rimpiazzare i tre bytes soppressi, ed in più effettuano un ulteriore controllo sui caratteri da inviare all'interprete.

Per la precisione il programma controlla se il carattere che sta per essere inviato all'interprete è un ←, nel qual caso si salta alla routine USER, mentre in caso contrario il controllo viene restituito all'interprete. La routine USER ha il compito di esaminare i caratteri che seguono la ← per individuare le differenti istruzioni (nel nostro esempio il comando è uno solo, ←C) e saltare alle corrispondenti subroutines.

Tenete presente che se il comando ha un solo argomento, come nel nostro esempio, per prelevare i caratteri che seguono occorre chiamare ogni volta la subroutine denominata NEXT.

Vi sono due routines del Basic che vengono usate nella routine ←C e in quella ERR. La prima interpreta 2 numeri o variabili separati dalla virgola, codificando il primo su 16 bit, il cui valore è posto alle locazioni $14 e $15 e il secondo su 8 bit, il cui valore è posto nel registro X.

La seconda di queste routine, se chiamata con un JMP provoca la stampa del messaggio ?ILLEGAL QUANTITY ER-

COMMODORE SCHOOL

ROR per poi restituire il controllo al sistema operativo.

Per finire una curiosità: i nuovi comandi vengono interpretati anche all'interno dei REM, per cui attenzione se in fase di debug REM-ate qualche linea.

N.B. Se usate un assemblatore il programma è rilocabile, a patto di cambiare manualmente i due bytes segnati con (R) nel listato Assembler. Questi devono infatti puntare sempre all'indirizzo corrispondente alla linea 73 del listato Assembler.

Esempio di inserimento in un programma Basic:

200 : ←C6,6:

Racchiudere, cioè, il comando tra due coppie di doppio punto.

Andrea & Alberto Boriani

```
10  "*$C000          ;ORIGINE = 49152
20  " LDA $4C        ; POSIZIONA
30  " STA (#115)     ;     IL
40  " LDA $17        ;(R) SALTO
45  " STA (#116)     ;     ALLE
50  " LDA $C0        ;(R) NUOVE
55  " STA (#117)     ; ROUTINES (LINEA 79)
56  " RTS
57  ";*********************************
58  "@NEXT           ;SUBROUTINE
60  " INC .(#122)  ; RICERCA
65  " BNE UNO        ;CARATTERE
70  " INC .(#123)
75  "@UNO
77  " RTS
78  ";*********************************
79  " JSR (NEXT)   ;INTERPRETA IL COMANDO
80  " TXA
90  " PHA ;    NELLO STACK
100 " LDX #0
110 " LDA /X(#122);PRELEVA IL CARATTERE
120 " CMP #95        ; E' UN '←' ?
130 " BEQ USER       ; SI',NUOVO COMANDO
135 "@END
140 " PLA ;RIPRISTINA X
145 " TAX
150 " JMP (#121)   ; NO,RITORNA AL BASIC
155 ";*********************************
160 "@USER           ;ROUTINES DI UTENTE
170 " JSR (NEXT)
180 " LDA /X(#122)
190 " CMP #67        ;E' UNA 'C'?
200 " BEQ C ;SI,SALTA
220 " JMP (END) ;NO,FINISCE
230 "@C      ;ROUTINE DI PROVA
240 " JSR (VAL)
250 " STA (#53281) ;SETTA SFONDO
252 " LDA ($14)
254 " STA (#53280) ;SETTA BORDO
255 " JSR (NEXT)
260 " JMP (END)
300 "@VAL
305 " JSR (NEXT)
```

COMMODORE SCHOOL

```
310 " JSR ($B7EB) ;VALUTA NUMERO SEGUENTE
315." TXA
340 " CMP #0          ;E' < DI'0' ?
350 " BMI ERR         ;SI',ERRORE
360 " CMP #16         ;E' > DI 15 ?
370 " BPL ERR         ;SI',ERRORE
371 " LDA ($14)
372 " CMP #0          ;E' < DI'0' ?
373 " BMI ERR         ;SI',ERRORE
374 " CMP #16         ;E' > DI 15 ?
375 " BPL ERR         ;SI',ERRORE
376 " TXA
380 " RTS
390 "@ERR ;ROUTINE DI ERRORE
400 " JMP ($B248) ;?ILLEGAL QUANTITY ERROR

READY.
```

ESPANSIONI COMANDI BASIC FIG.1

```
100 REM ********************************
110 REM * EXTRA BASIC PER COMMODORE 64 *
120 REM *    ALBERTO & ANDREA BORIANI   *
130 REM * VIA ACACIE 3 CUSANO MILANINO *
140 REM *         TEL.02/6195626        *
150 REM ********************************
160 :
170 REM **** LETTURA DATA ****
180 :
190 FOR F=0 TO 101: READ D: POKE 49152+F,D: C=C+D: NEXT
200 REM ****   CHECKSUM   ****
210 IF C<>10299 THEN PRINT"ERRORE NEI DATA": END
220 SYS 49152
230 REM **** DATI ****
240 :
250 DATA 169,76,141,115,0,169,23,141,116
260 DATA 0,169,192,141,117,0,96,230,122,208
270 DATA 2,230,123,96,32,16,192,138,72,162
280 DATA 0,161,122,201,95,240,5,104,170
290 DATA 76,121,0.32,16,192,161
300 DATA 122,201,67,240,3,76,36,192,32
310 DATA 71,192,141,33,208,173,20,0,141,32
320 DATA 208,32,16,192,76,36,192,32,16
330 DATA 192,32,235,183,138,201,0,48
340 DATA 17,201,16,16,13,173,20,0,201,0
350 DATA 48,6,201,16,16,2,138,96,76,72,178
360 :
370 REM ESEMPIO SINTASSI:
380 REM       ←C6,6
390 REM COLORA FONDO E BORDO IN BLU

READY.
```

ESPANSIONI COMANDI BASIC FIG.2

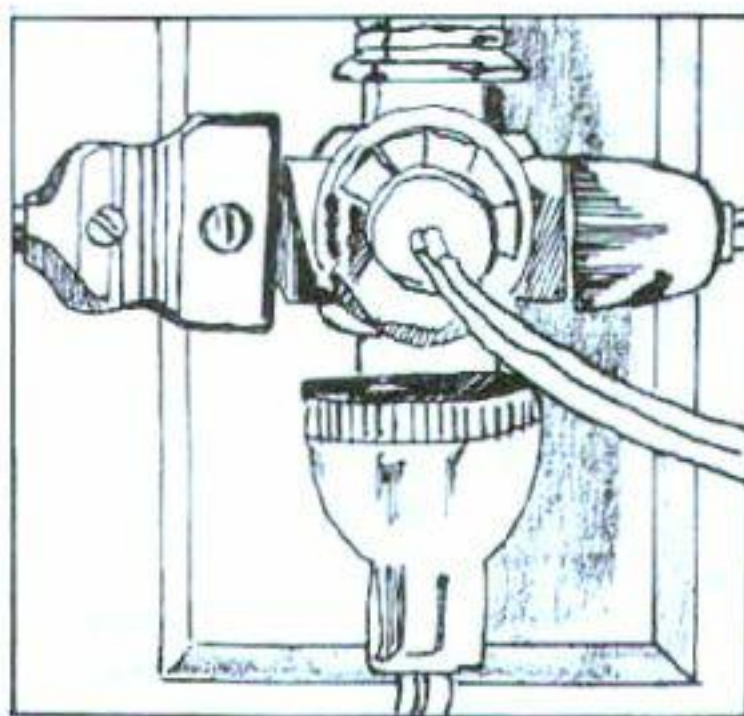
LA GESTIONE DELL'INTERRUPT NEL COMMODORE 64

Nuove, straordinarie possibilità d'impiego per il vostro computer.

Il lettore dovrebbe sapere che all'interno dei computer Commodore c'è un orologio, con ciclo di 24 ore, che inizia a contare il tempo dall'istante in cui viene data tensione all'apparecchio. Tale contatore di tempo è piuttosto preciso, dato che è controllato da un quarzo e gli sono dedicate due variabili. Una di queste è il TI, variabile riservata al calcolatore nel senso che non può essere modificata dall'utente. Se infatti tentiamo di digitare TI=43 otteniamo un SYNTAX ERROR. La seconda variabile è invece TI$ e serve per regolare l'orologio sull'ora desiderata.

Esempio: TI$= "121500" comunicherà al calcolatore che è mezzogiorno e un quarto. E' ovvio che TI$ viene aggiornata di continuo, ed esattamente 60 volte al secondo, dalla variabile TI. Quest'ultima variabile ha riservate tre locazioni di memoria, 160, 161, 162, nel Commodore 64 come nel Vic 20. Il loro contenuto viene incrementato, l'uno dopo l'altro, da zero a 255. Il microprogramma di figura 1 è di aiuto per comprendere quanto asserito.

Ogni sessantesimo di secondo si deve, pertanto, verificare un processo per cui, anche se l'utente non lo richiede esplicitamente, i contenuti delle tre locazioni, vengono aggiornati, elaborati e trasferiti in TI$ se richiesto.

A tutto questo, e a tante altre cose, provvede il Sistema Operativo (S.O.) e vedremo in che modo.

Pochi sanno che il 6502 e il 6510, microprocessori utilizzati sul Vic 20 e sul Commodore 64, hanno, tra gli altri, un piedino (pin) ed esattamente il N.4 (6502) ed il N.3 (6510) chiamato IRQ (Interrupt ReQuest = richiesta di interruzione di programma). In assenza del segnale appropriato sul pin specificato, il microprocessore lavora normalmente eseguendo il programma allocato in memoria. Non appena invece giunge l'impulso, il microprocessore, automaticamente, compie le seguenti operazioni:

- Porta a termine l'istruzione che sta eseguendo in quel momento (attenzione: la singola istruzione in Linguaggio Macchina e non l'eventuale istruzione Basic).
- Salva (memorizza) il contatore di programma, cioè l'indirizzo dell'istruzione successiva, che avrebbe eseguito se non vi fosse stata la richiesta di interruzione.
- Salva il registro di Flag di stato (Carry, Overflow, Negativo, ecc.) in un modo che in questa sede non interessa.
- Salta ad eseguire il programma in Linguaggio Macchina il cui indirizzo di partenza è contenuto nelle locazioni FFFE (parte bassa) e FFFF (parte alta). Si tenga presente che il modo di operare descritto è una caratteristica del microprocessore e non del computer. Qualunque calcolatore basato sul 6502 oppure sul 6510 terrà conto del contenuto degli indirizzi FFFF FFFF alla richiesta di un interrupt.
- Il nuovo programma in L.M. deve terminare con una istruzione RTI (ReTurn from Interrupt = ritorna dall'interruzione), codice esadecimale 40, istruzione implicita ad un solo byte.
- Quando il microprocessore incontra l'istruzione RTI, esso richiama nuovamente il registro dei Flag interni precedentemente memorizzati e l'indirizzo dell'istruzione che avrebbe dovuto eseguire se non vi fosse stata richiesta di interruzione.

A questo punto il micro è nelle stesse condizioni in cui era quando si è verificato l'IRQ. Attenzione: l'accumulatore, il registro X e quello Y non sono più gli stessi di prima, ma assumono l'eventuale valore derivante dalla elaborazione della routine di interrupt.

Vediamo ora a che può servire quanto detto finora. Il piedino del circuito integrato specificato è connesso con un clock (segnale temporizzato) per cui il computer (badate bene: ogni sessantesimo di secondo), interrompe l'elaborazione ed esegue alcune routine di controllo. Per esempio, non appena si accende l'apparecchio, ed anche in molti altri casi, il cursore lampeggia con un ritmo ed in una posizione determinati proprio dalla routine di interrupt. Questa esamina, ad altissima velocità, se è stato digitato un tasto oppu-

re se è in corso uno scambio di dati tra computer e periferiche (registratore, floppy, stampante, penna luminosa ecc.). Se tutto ciò fornisce un risultato negativo va a controllare quanto tempo è passato dalla precedente interruzione e, a seconda dei casi, cancella o visualizza il cursore in modo da farlo lampeggiare e aggiorna TI!!!

Quando ad esempio facciamo girare un programma Basic la cui durata di elaborazione apparente è di un solo minuto, la routine di interrupt sarà stata interrogata circa 3600 volte (60 x 60).

```
100 PRINT PEEK(160)PEEK(161)PEEK(162): GOTO 100
```

Figura 1

Quanto detto finora, benchè, ce lo auguriamo, sia interessante, può sembrare senza applicazioni pratiche dato che le locazioni di memoria che contengono l'indirizzo di IRQ sono in un blocco di memoria ROM e quindi non alterabili da parte dell'utente. Ciò è vero a livello di architettura del sistema 65XX, ma bisogna tener presente che la routine di interrupt, tra le altre cose, salta ad eseguire, ad un dato punto, una routine il cui indirizzo di partenza è contenuto nelle due locazioni RAM 788-789 (esadecimale = 0314-0315) sia per il Vic 20 che per il Commodore 64. Poichè la memoria RAM è alterabile da parte dell'utente, ne deriva che una manipolazione delle due locazioni apre nuove, straordinarie possibilità.

Il diagramma di flusso di figura 2 dovrebbe chiarire le idee su quanto detto finora. Si noti, tra l'altro, la nuova istruzione a tre bytes (6C LA HA) che viene così interpretata:

Vai ad eseguire la routine il cui indirizzo effettivo è contenuto nelle due locazioni che seguono (LA HA).

Prima di andare avanti con i consueti esempi pratici, parliamo di altre istruzioni in codice macchina che riguardano l'interrupt, iniziando da SEI (SEt Interrupt) codice macchina 78, istruzione implicita ad un solo byte. Abbiamo detto che ogni sessantesimo di secondo il clock genera un impulso che fa scattare la routine di interrupt. Questa ha una durata inferiore al sessantesimo di secondo (figura 3A) e quando restituisce il controllo alla routine principale, il segnale presente al pin dell'IRQ è tale per cui effettivamente il microprocessore riprende a lavorare nel punto in cui aveva smesso al momento della richiesta di interrupt. Supponiamo ora che la routine di interrupt (richiamata all'istante T1 di figura 4), abbia una durata superiore al ciclo di clock: mentre la routine è in piena attività il microprocessore riceve sul pin interessato un nuovo segnale di interruzione all'istante T2. «Crederà», di conseguenza, che sia stata generata una seconda routine di interrupt e va ad eseguirla. Analoghi malfunzionamenti si verificano se nel preciso istante del ritorno dalla routine, il segnale sul pin diventa attivo: si rischia, infatti, di far girare solo la routine di interrupt e mai quella «principale».

Si presenta allora la necessità di ignorare, in alcuni casi, la richiesta di interrupt. L'istruzione implicita ad un solo byte che consente tale possibilità è CLI, codice macchina 58, che viene così interpretata:

A partire da questo istante ignora qualsiasi richiesta di interruzione che venga effettuata.

Abbiamo pertanto a disposizione quello che in gergo viene chiamato interrupt mascherabile. L'istruzione che consente di tener conto nuovamente della richiesta IRQ è SEI, codice operativo 78, implicita anch'essa. Essa viene così interpretata:

A partire da questo istante devi servire ogni nuova richiesta di IRQ.

Si tenga ben presente la sostanziale differenza tra CLI ed RTI. La prima consente di tener conto di richieste IRQ: quando cioè il segnale adatto è presente al

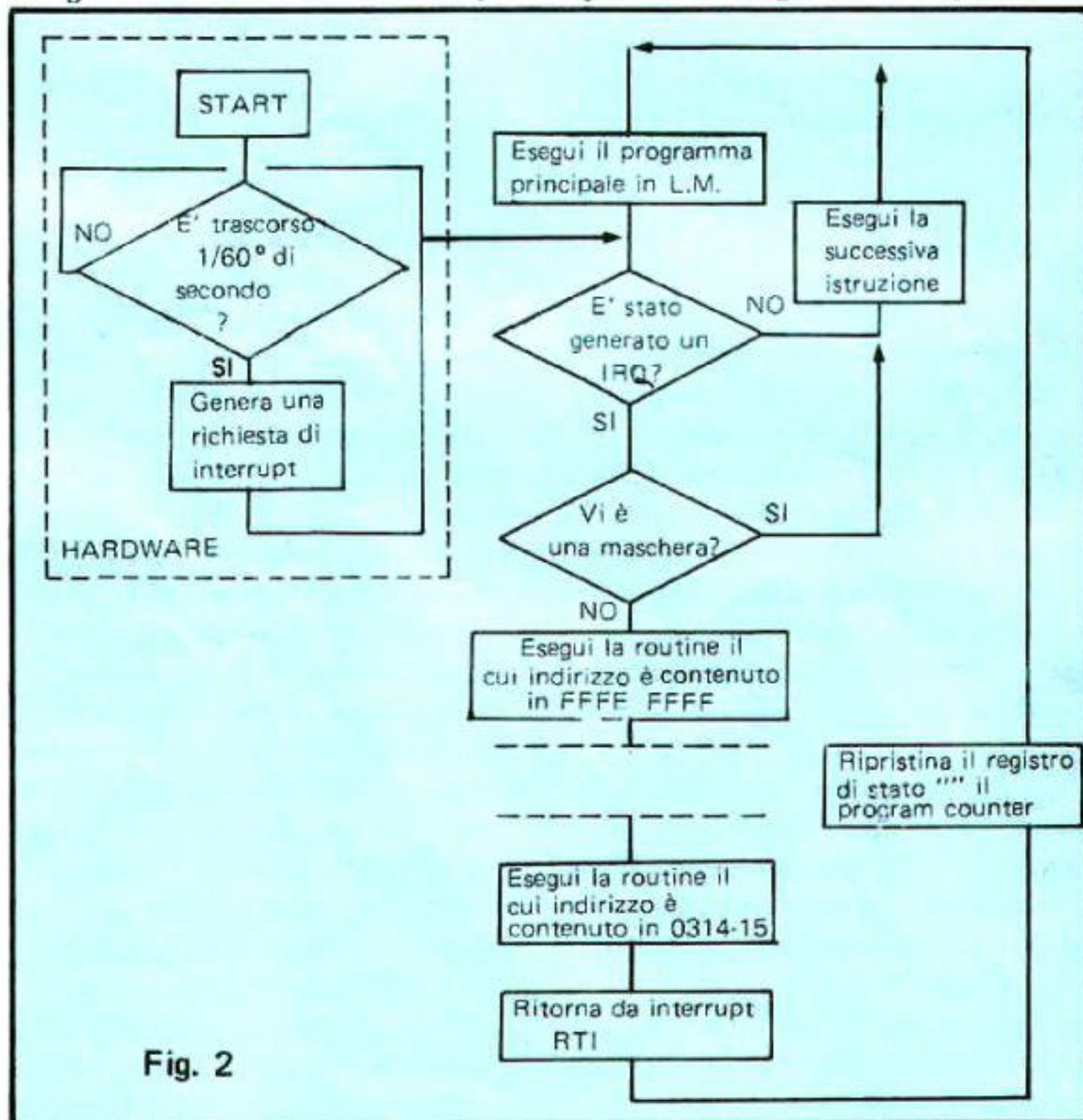



Fig. 2

pin, scatterà la routine il cui indirizzo di partenza è localizzato in FFFE ed FFFF. Con RTI, invece, tale routine termina e si ha un ritorno al programma principale.

Vediamo ora un'applicazione pratica. Utilizzando un monitor per linguaggio macchina oppure il potente programma pubblicato in questo numero (IPER MONITOR) richiamate la memoria in modo da esaminare i valori contenuti nelle locazioni 788-789 (esa 0314-0315). In queste noteremo i valori 31 EA nel Commodore 64 e i valori (decimali) 191 e 234 nel Vic 20.

Richiamiamo ora una zona di memoria alterabile a piacere, come quella relativa al buffer della cassetta (033C-03FB) e digitiamo, a partire appunto da 033C il programma che segue (Commodore 64):
A9 00 8D 00 04 A9 00 8D 00 D8 60.

Lanciando questo programma, da monitor o da Basic mediante SYS 828, si noterà l'apparizione della chiocciolina, in nero, nella prima locazione di schermo in alto a sinistra, Se però cancelliamo il video, col tasto Clr/Home, la chiocciolina scompare (elementare...). Richiamiamo ora le locazioni precedentemente trascritte e modifichiamole come segue:
A9 00 8D 00 04 A9 00 8D 00 D8 4C 31 EA

Subito dopo chiediamo il contenuto delle locazioni di memoria 0314-0315 e modifichiamo il contenuto in modo che esso «punti» alla nostra routine 3C 03. Non appena battiamo il tasto Return apparirà la chiocciola nera in alto a sinistra. Provate ora a cancellare lo schermo col tasto Clr/Home: la chiocciolina è sempre in alto e non viene cancellata con nessun mezzo! Che cosa è successo?

Ogni sessantesimo di secondo il clock chiama la routine di interrupt; il Sistema Operativo, durante tale operazione, va ad eseguire anche una routine il cui indirizzo è contenuto in 0314-0315 che da noi è stato modificato in 033C. Pertanto la routine di interrupt, prima di andare ad eseguire il «solito» giro che inizia a partire da EA31, è costretta a passare per la nostra routine che, come abbiamo visto, provvede a visualizzare la chiocciolina nera in alto a sinistra.

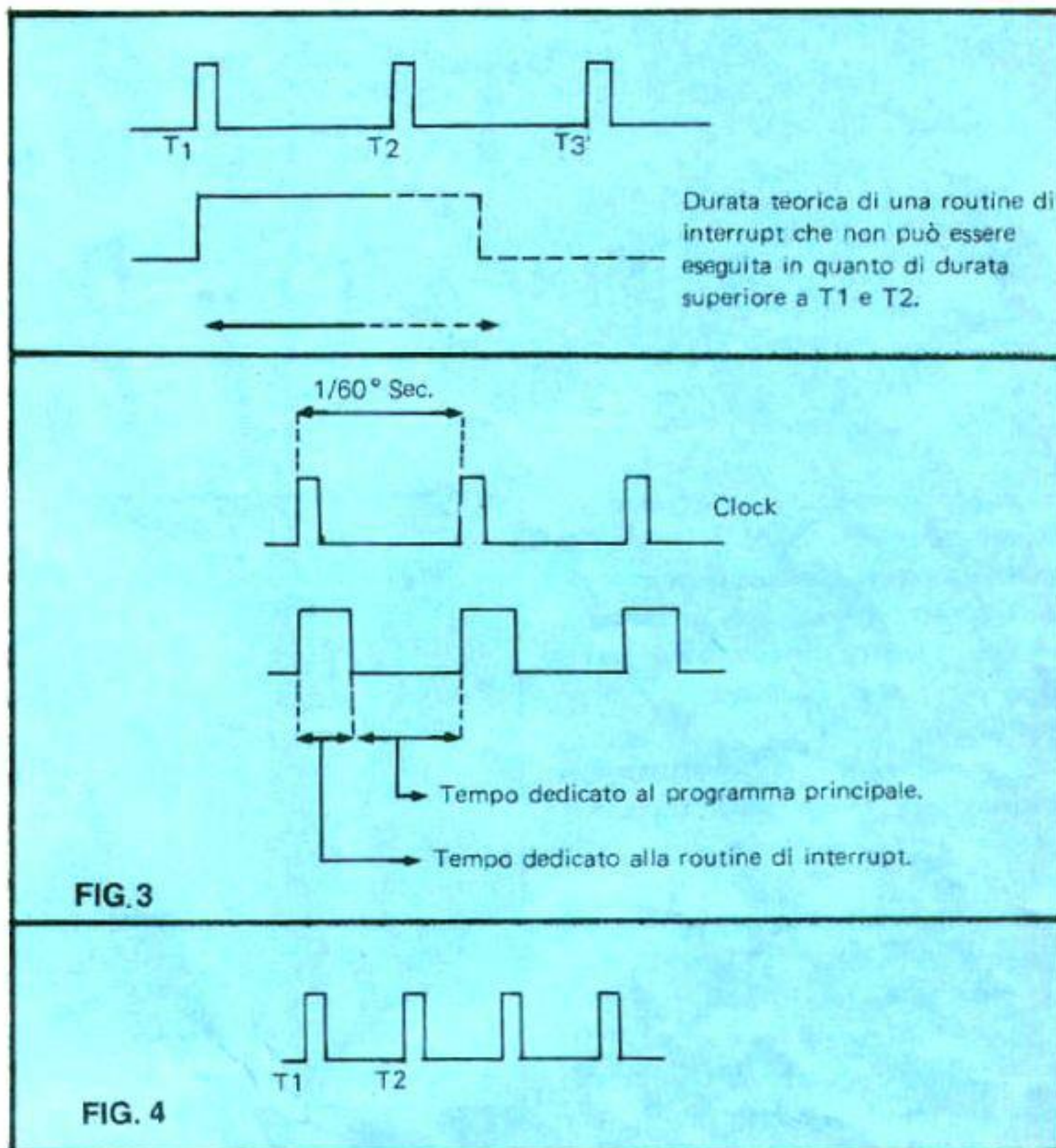



FIG. 3

FIG. 4

Il tasto Clr/Home, pertanto, funziona perfettamente. La diversità consiste nel fatto che un sessantesimo di secondo dopo che lo schermo viene cancellato, la chiocciolina viene ristampata nuovamente dalla routine di interrupt da noi modificata.

Il lettore che desideri sperimentare la tecnica descritta in questo articolo, tenga presente che è indispensabile provare dapprima la routine che si desidera inserire, come una normale routine in L.M. (che termina con RTS, 60). Solo in seguito, accertatone il funzionamento, sostituire 60 con 4C 31 EA e modificare i contenuti di 0314-0315. Nel caso in cui si desideri modificare una routine già inserita è *indispensabile:*

- Riscrivere in 0314-0315 i valori 31 EA.
- Modificare la nostra routine in modo che termini con RTS (60).
- Sperimentare la nuova routine come una normale routinetta (SYS...)
- Modificare 60 con 4C 31 EA.
- Modificare i contenuti di 0314-0315 in modo che «puntino» alla nostra routine.

Non vi azzardate a modificare le locazioni del vettore di interrupt con dei semplici POKE o comunque da Basic!!! Mentre, infatti, l'istruzione POKE relativa alla parte bassa dell'indirizzo viene eseguita, può arrivare al pin una richiesta di interruzione che farebbe partire la routine con l'indirizzo modificato a metà...

Si tenga ben presente che in ogni caso, premendo Run/Stop e Restore, si ottiene il ripristino dei valori originali del vettore di Interrupt.

Alesssandro de Simone

INTERRUPT: APPLICAZIONI PRATICHE

Vediamo ora alcune applicazioni col Commodore 64 che hanno il solo scopo di studiare il comportamento del microprocessore. Niente di male che, lavorando di fantasia, il lettore «inventi» nuove possibilità per generare videogiochi o altre applicazioni.

• *Colore*. Facendo partire il programma vi accorgerete che ogni volta che premete un tasto i colori del bordo e del fondo cambiano.

• *Sprite*. Uno sprite colorato percorre il perimetro dello schermo TV mentre girano altri programmi oppure mentre listate.

• *Suono*. Come per la prima routine, solo che si ode un suono casuale tutte le volte che viene battuto un tasto.

• *Flashing*. Questa è forse l'applicazione più interessante. Come è noto alcuni computer hanno non solo la possibilità di far apparire dei messaggi in reverse, come i Commodore, ma anche lampeggianti (Flashing). Con la routine pubblicata è possibile ottenere tale possibilità col Commodore 64. Se, ad esempio, volete far apparire lampeggiante il messaggio:

QUESTA E' UNA PROVA
digitate così:

PRINT CHR$(160) "QUESTA E' UNA PROVA" CHR$(160)

Il carattere CHR$(160) rappresenta quello ottenibile premendo contemporaneamente il tasto Shift e la barra spaziatrice. In altre parole, con la routine Flashing, appariranno lampeggianti tutti i messaggi che iniziano e terminano con il carattere (160).

```
3 REM  COLORE NELL'INTERRUPT
4 REM  BY MARIANI GIANCARLO
5 REM  VIALE BRIANZA 72-20036 MEDA (MI)
6 REM  TEL. 0362/72565
7 REM
8 REM  PER FERMARE: RUN STOP/RESTORE
9 REM
10 A=49152:REM A INTERAMENTE RILOCABILE
20 B=0:FORT=ATOA+29:READC:POKET,C:B=B+C:NEXT
30 IFB<>3119THENPRINT"■ERRORE NEI DATI":END
35 B=INT(A/256):C=A-B*256
40 POKEA+2,C+13:POKEA+7,B
45 SYSA
1000 DATA 120 , 169 , 13 , 141 , 20
1005 DATA 3 , 169 , 32 , 141 , 21
1010 DATA 3 , 88 , 96 , 165 , 197
1015 DATA 201 , 64 , 240 , 8 , 141
1020 DATA 32 , 208 , 105 , 1 , 141
1025 DATA 33 , 208 , 76 , 49 , 234

READY.
```

```
1 REM **************************
2 REM *  FLASHING   ROUTINE     *
3 REM *           BY            *
4 REM * F.SORGATO & G.MARIANI   *
5 REM *       VIA L.RHO 62      *
6 REM *     20036 MEDA (MI)     *
7 REM *     TEL.0362/70598      *
8 REM **************************
10 REM*   SYS 40704 ABILITA     *
11 REM*  SYS 40820 DISABILITA   *
12 REM*     LA LOCAZIONE 680    *
```

```
13 REM* CONTIENE LA FREQUENZA *
14 REM*      DEL LAMPEGGIO      *
15 REM************************
16 REM
17 REM
100 POKE56,159:CLR:REM TOP MEMORY
110 FORT=40704TO40831:READA:H=H+A
120 POKET,A:NEXT
130 IFH<>14174THENPRINT"ERRORE NEI DATA": END
1000 DATA169,0,133,2,120,169,17,141
1010 DATA20,3,169,159,141,21,3,88
1020 DATA96,165,2,201,5,208,22,238
1030 DATA167,2,173,167,2,205,168,2
1040 DATA208,74,162,0,142,166,2,134
1050 DATA2,202,142,167,2,166,2,189
1060 DATA96,159,133,168,189,101,159,133
1070 DATA167,160,0,177,167,201,96,208
1080 DATA9,173,166,2,24,105,128,32
1090 DATA111,159,173,166,2,201,128,208
1100 DATA7,177,167,24,105,128,145,167
1110 DATA200,192,200,208,222,76,106,159
1120 DATA4,4,5,6,7,0,200,144
1130 DATA88,32,230,2,76,49,234,141
1140 DATA166,2,200,96,120,169,49,141
1150 DATA20,3,169,234,141,21,3,96
1160 :
1170 PRINTCHR$(160)" PROVA "CHR$(160):END

READY.
```

```
3 REM  SPRITE NELL'INTERRUPT
4 REM  BY MARIANI GIANCARLO
5 REM  VIALE BRIANZA 72
6 REM  20036 MEDA (MI)  T.0362/72565
7 REM
8 REM  PER FERMARE:RUN STOP/RESTORE
9 REM
10 A=49152:REM A INTERAMENTE RILOCABILE
20 B=0:FORT=ATOA+133:READC:POKET,C:B=B+C:NEXT
30 IFB<>17709THENPRINT"ERRORE NEI DATI":END
35 B=INT(A/256):C=A-B*256
40 POKEA+35,C+46:POKEA+40,B
50 SYSA
1000 DATA 169 , 0 , 133 , 252 , 133
1005 DATA 251 , 169 , 255 , 162 , 0
1010 DATA 157 , 192 , 2 , 232 , 224
```

```
1015 DATA 63 , 208 , 248 , 169 , 11
1020 DATA 141 , 248 , 7 , 169 , 1
1025 DATA 141 , 21 , 208 , 169 , 7
1030 DATA 141 , 39 , 208 , 120 , 169
1035 DATA 46 , 141 , 20 , 3 , 169
1040 DATA 32 , 141 , 21 , 3 , 88
1045 DATA 96 , 165 , 252 , 201 , 2
1050 DATA 240 , 50 , 201 , 3 , 240
1055 DATA 61 , 201 , 0 , 240 , 7
1060 DATA 201 , 1 , 240 , 23 , 76
1065 DATA 49 , 234 , 169 , 30 , 141
1070 DATA 0 , 208 , 238 , 1 , 208
1075 DATA 173 , 1 , 208 , 201 , 224
1080 DATA 208 , 2 , 230 , 252 , 76
1085 DATA 49 , 234 , 238 , 0 , 208
1090 DATA 173 , 0 , 208 , 201 , 254
1095 DATA 208 , 2 , 230 , 252 , 76
1100 DATA 49 , 234 , 206 , 1 , 208
1105 DATA 173 , 1 , 208 , 201 , 55
1110 DATA 208 , 2 , 230 , 252 , 76
1115 DATA 49 , 234 , 206 , 0 , 208
1120 DATA 173 , 0 , 208 , 201 , 32
1125 DATA 208 , 4 , 169 , 0 , 133
1130 DATA 252 , 76 , 49 , 234

READY.
```

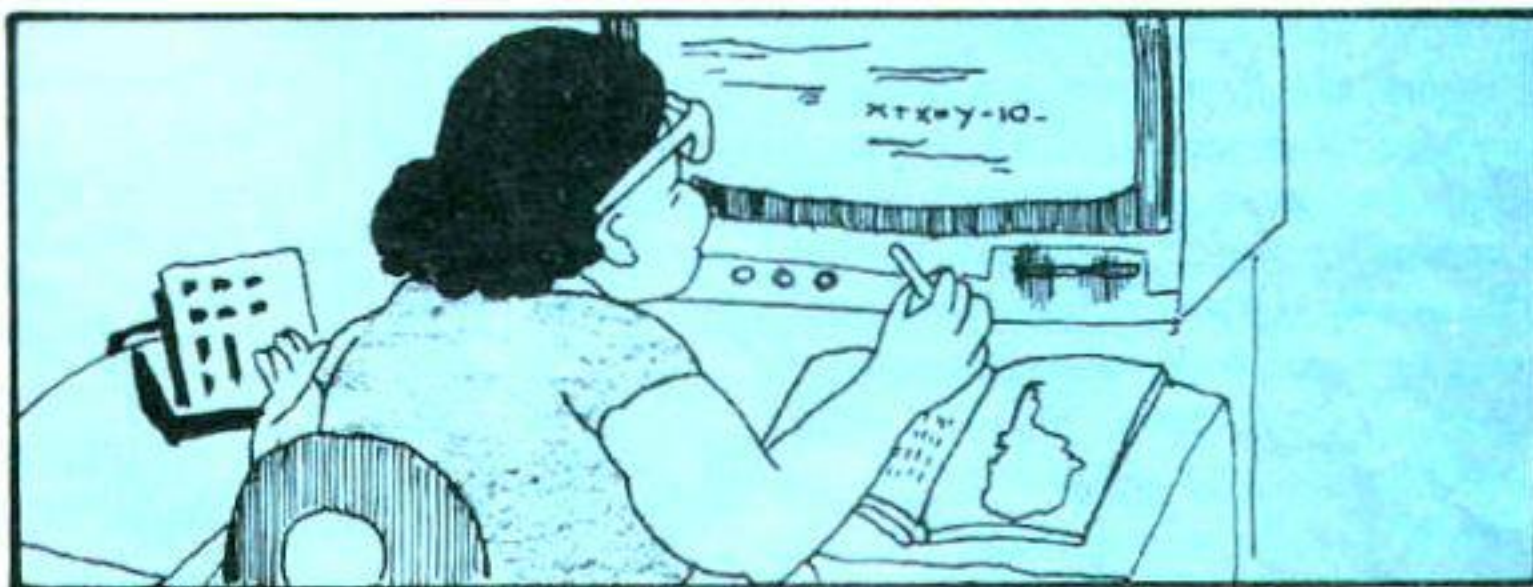
```
1 REM    SUONO NELL'INTERRUPT
2 REM     GIANCARLO MARIANI
3 REM     VIALE BRIANZA 72
4 REM     20036 MEDA (MI)
5 REM
6 REM "B" INTERAMENTE RILOCABILE
7 REM
8 B=49152: B1=B+18
10 FOR K=0 TO 62:READ A:POKE B+K,A
20 X=X+A:NEXT:IF X <>7159 THEN PRINT"ERRORE NEI DATA": END
30 B2=INT(B1/256):B3=B1-B2*256
40 POKE 251,B3:POKE 252,B2
50 SYS B
1000 DATA 169,10,141,24,212,120,165,251,141,20,3
1010 DATA 165,252,141,21,3,88,96,165,197,201,64
1020 DATA 240,36,133,2,169,0,141,4,212,141,5
1030 DATA 212,141,6,212,169,25,141,5,212,169,0
1040 DATA 141,6,212,169,17,141,4,212,165,2,141
1050 DATA 1,212,141,0,212,76,49,234

READY.
```

LINGUAGGIO MACCHINA

(TERZA PUNTATA)

Finalmente il seguito dell'argomento iniziato nei numeri 6 e 8 di Commodore Computer Club.

Ora possiamo "scrivere" un numero intero compreso tra -32768 e +32767. Vediamo ora come eseguire la somma fra due numeri decimali e poi esadecimali. Facendo la somma dell'esempio qui sotto riportato, notiamo che quando la somma di due cifre corrispondenti supera il numero nove, dobbiamo considerare il riporto nella successiva colonna (somma tra 8 e 5; tra 6 e 4).

Decimale	Esadecimale
1 1 Riporto	1 Carry
16280 +	1A03 +
4250 =	9B77 =
20530	B57A

Analogamente avviene per due numeri esadecimali, solo che consideriamo il riporto quando la somma supera il numero F (15); (esempio: somma fra A e B della figura sopra riportata). Prima di continuare ricordiamo che, in inglese, riporto si traduce col termine CARRY, ed in seguito lo chiameremo sempre in questa maniera. Tra le istruzioni in L.M. del 6502 ve ne sono ben otto di somma tra due byte, e ne esaminiamo ora una.

Codice esadecimale 69: istruzione a due byte; codice mnemonico ADC≠ derivato da ADd, accumulator immediate with Carry.
Esempio:

LM : 69 72
Ass : ADC≠ $72

Quando il microprocessore incontra questa istruzione, esegue la somma tra il valore che si trova in quel momento nell'acculatore, l numero esadecimale 72 e l'eventuale Carry.

Vediamo ora alcuni esempi:

```
LM              ASSEMBLER

                @826 = START
033A 18         CLC
033B A9 02      LDA# #2
033D 69 07      ADC# #7
033F 8D 45 03   STA  QUI
0342 60         RTS
.....EA         NOP
.....EA         NOP
0345 EA         NOP
     EA              = QUI
     EA         NOP
     EA         NOP
```

COMMODORE SCHOOL

REMarks

033A 18 Clear Carry (CLC)

Istruzione ad un solo byte, implicita, cioè non ha bisogno di indirizzi o di dati per la sua interpretazione. Questa istruzione cancella il Carry eventualmente presente nel registro della CPU. Ritorneremo in seguito su questa istruzione.

033B A9 02 LDA 2. Vedi numero 6 di Commodore Computer Club.

0341 60 RTS Return subroutine.

Istruzione implicita anche questa: non c'è bisogno di altre informazioni per specificare il compito da svolgere.

0342 EA...EA NOP No OPeration.

Trascriviamo questo gruppo di EA perchè in seguito sia più semplice rintracciare sul video il risultato della somma. Per la cronaca EA è un'istruzione implicita che significa: "Non eseguire alcuna operazione"; al contrario di quanto possa sembrare è un'istruzione molto utile, come vedremo in seguito. Per far girare il programma battiamo di seguito:

POKE 828,2: POKE 830,7: SYS :826)

Utilizziamo un monitor, come l'iper monitor pubblicato in questo numero:

Battiamo: M 033A 034A apparirà:

```
.M 033A  034A
.: 033A  18 A9 02 69 07 8D 45 03
.. 0342  60 EA EA 09 EA EA EA ..
.X
READY
■
```

Ora, sia modificando il monitor, sia ricorrendo a POKE 828 e POKE 830, modifichiamo il valore di 033C e di 033E e, ripetendo le operazioni precedenti, in 0345 comparirà sempre il risultato della somma, a meno che la somma dei due numeri caricati non superi il valore FF; infatti, consideriamo le seguenti somme esadecimali:

```
          NB -  0C  + 04  = 10
          NB -  40  + 0A  = 4A
      invece:   0B  + 03  = 0E
Infatti:        FC  + 14  = 10
   0C  = 12 dec.
   04  = 04 dec. AB + 63  = 0E
   04 + 12 = 16 dec. = 10 esa.
```

Come si può notare, allo stesso risultato si può arrivare anche sommando FC e 14. Come facciamo ora a sapere se il valore contenuto in 0345 è il risultato di una somma inferiore o superiore ad FF? A tale scopo consideriamo il registro di stato (ST) della CPU 6502: esso è un particolare registro di sedici bit, interno alla CPU stessa, che memorizza diverse informazioni tra le quali il Carry. Il bit dell'ST corrispondente al Carry, viene automaticamente posto (settato) ad 1 se, dopo l'esecuzione di alcune istruzioni, tra le quali la somma, si supera il valore FF. Da quel momento esso non viene più cancellato (cioè resettato o portato a 0), anche se, in seguito, si esegue una somma che non ha riporto.

Ecco perchè la prima istruzione di una procedura di somma deve SEMPRE essere CLC, che viene così interpretata: resetta il Carry che eventualmente è stato posto ad 1 da una operazione precedente.

Miglioriamo ora il programma in L.M. visto e, per fare questo, esaminiamo un'altra istruzione di BRANCH (salto condizionato):

BCS: Branch on C Set to 1 = Salta se il Carry è uguale ad 1.

Codice operativo: B0 xx; dove xx rappresenta l'entità del salto, vedi pag. 49 numero 6 C.C.C. (REMarks 034C).

```
                  0826 = START
033A 18           CLC
     A9 00        LDA# #0
     8D 52 03     STA: RIPORTO
     A9 02        LDA# #2
     69 07        ADC# #7
     8D 45 03     STA: BYTE SOMMA
     B0 01        BCS + MODIFICA
     60           RTS
034A              = MODIFICA
```

```
                  LDA# #1
     8D 52 03     STA: RIPORTO
     60           RTS
     EA           NOP
.... ..           ....
     EA           = RIPORTO
     EA           = BYTE SOMMA
     EA           NOP
```

REMarks

A9 00

8D 52 03 - Queste due istruzioni scrivono 00 nella locazione 0352

che servirà a visualizzare l'eventuale riporto.

B0 01 - Se la somma appena eseguita supera il valore FF, il bit di Carry viene posto a 1, e pertanto il salto di un byte è eseguito e l'elaborazione continua da 034A.

60 - Se la somma eseguita non ha riporto, il Carry rimane come prima, e cioè a 0 (cfr. ist. 033A 18); si ritorna al BASIC.

A9 01

8D 52 03 - Nella locazione 0352 indicata in Assembler con RIPORTO, viene trascritto il valore 01.

Facciamo ora girare il programma nel solito modo (SYS 826), ricordando che i nuovi indirizzi della POKE sono 833 e 835. Se la somma eseguita avrà riporto o meno lo noteremo dalla presenza o meno, di 01 in 0352.

Dato che ci siamo, commentiamo alcune righe del programma scritto in Assembler.

826 = START - Comunica al programma Assembler, definendola con il nome "START", la locazione dalla quale il programma in L.M. deve essere allocato. Il programma (Assembler) automaticamente provvederà ad incrementare gli indirizzi per le successive istruzioni. Ciò significa che se, per esempio, vogliamo collocarlo nella memoria a partire dalla locazione 0355, invece che dalla 033A, sarà sufficiente modificare solamente $0355 = START; a modificare tutti gli indirizzi provvederà automaticamente l'Assembler.

STA: RIPORTO. Memorizza il valore attuale (corrente) dell'accululatore nella locazione indicata in seguito con «riporto».

STA: BYTE SOMMA. Trasferisce il valore corrente dell'accululatore nel byte definito in questo modo.

BCS — MODIFICA. Se il Carry è uguale a 1, il programma continua dalla locazione indicata come "MODIFICA". Attenzione: l'Assembler elabora automaticamente l'entità del salto, evitando noiosi calcoli. I vari nomi di fantasia, che facilitano nettamente la stesura di un programma, sono chiamati LABELS (etichetta).

Come si può notare, il programma scritto in Assembler è di gran lunga più semplice da interpretare dell'L.M. Eseguiamo ora una somma di due numeri, ciascuno dei quali occupa due byte.

```
                 Decimale
2CEA  +
3221  =
-------
5F1B
                 11290  +
                 14891  =
Esadecimale      -------
                 26181
```

In pratica, dobbiamo dapprima eseguire la somma tra gli ultimi due byte (EA — 21), trascrivere il risultato da qualche parte e tener presente l'eventuale Carry (nel caso specifico esiste). In seguito, sommiamo i primi due byte fra loro e l'eventuale Carry (2C — 32 — 1). Il programma può essere il seguente:

```
CLC
LDA# #SEA
ADC# #$ 21
STA: LSB
LDA# #$ 2C
ADC# #$ 32
STA: MSB
RTS
= LSB
= MSB
```

Prima di eseguire la seconda somma (LDA $2C - ADC → —$32) NON bisogna inserire CLC, perchè e necessario considerare il Carry, eventualmente presente in seguito alla prima somma. Il lettore, per esercizio, può modificare il programma inserendo una «spia» per l'eventuale secondo riporto, come abbiamo fatto nel programmino precedente. Prima di terminare parliamo di un'altra istruzione implicita: SED (SEt Decimal mode), codice operativo F8. Dall'istruzione successiva a SED le somme vengono eseguite in decimale anzichè in esadecimale: F8 è di notevole comodità, in molti casi: provate ad inserirla nei programmi precedenti prima di CLC o subito dopo. Attenzione però che, se il modo di operare è decimale e chiediamo di sommare valori esadecimali, si verificano degli errori. Per ripristinare in modo esadecimale è necessario l'istruzione implicita CLD (CLear Decimal mode), codice operativo D8; senza questa istruzione infatti la CPU continuerà sempre a ragionare in decimale. Consiglio a chi desidera sperimentare l'istruzione SED, di inserire D8 prima di ogni RTS per evitare malfunzionamenti del Basic.

Per concludere, a partire dalla locazione 033A, inserite il seguente programma e battere SYS(826), dopo aver cancellato lo schermo.

```
033A A2  00  8D  4D  03  8D  00  04
0342 A9  00  9D  00  D8  E8  E0  18
034A D0  F0  60  43  4F  4D  4D  4F
0352 44  4F  52  45  20  43  4F  4D
035A 50  55  54  45  52  20  43  4C
0362 55  42  20  2A  2A  00  00  00
```

Dato che contiene istruzioni che non abbiamo ancora incontrato, lo commenteremo nella prossima puntata e non vi diciamo a cosa serve per incuriosirvi. Il lettore tenga comunque presente che 034D contiene l'inizio; da 034D a 036... si può modificare il contenuto senza alcun pericolo... che cosa aspettate a decifrare quest'ultima frase?

Alessandro de Simone

SEGUE →

COMMODORE SCHOOL

Precisazioni

I due articoli sull'Assembler e linguaggio macchina finora pubblicati (vedi C.C.C. N. 6 e 8) erano riedizioni di scritti precedenti. In tali articoli gli esempi cui si fa riferimento sono validi per i computer Commodore della serie 2000, 3000, 4000 e 8000. Nonostante i «ragionamenti» seguiti siano gli stessi anche per il Vic 20 e per il Commodore 64, i listati in Assembler e in L.M. pubblicati non possono girare su tali macchine. Infatti i listati cui si fa riferimento agiscono tutti sull'area di schermo. Si sa inoltre che mentre per i vecchi PET non è necessario colorare la cella video, tale operazione risulta indispensabile per i modelli Vic 20 e CBM 64. In effetti gli adattamenti a questi computer li avrebbero dovuti fare i lettori sia per esercitarsi sia per "inventarsi" alcune routine. Numerosi lettori, invece, ci hanno telefonato pregandoci di pubblicare gli adattamenti per le macchine in loro possesso.

Pur non avendo alcuna difficoltà ad accontentarli, ricordiamo che la padronanza di un settore così come il Linguaggio Macchina si può raggiungere a patto di lavorare con impegno da soli, realizzando adattamenti e non... chiedendoli alla Redazione delle riveste specializzate.

```
100 PRINT"SYS 826 FA APPARIRE IN ALTO"
110 PRINT"A SINISTRA SULLO SCHERMO DEL"
120 PRINT"COMMODORE 64 UN CARATTERE (X)"
130 PRINT"SELEZIONABILE CON POKE 827,X."
140 PRINT"COLORE (Y) SELEZIONABILE CON:"
150 PRINT"POKE 832,Y"
160 AD=826:AD$="SYS 826"
170 :
180 READ A$:IFVAL(A$)<0THENPRINT""AD$:END
190 X1=ASC(LEFT$(A$,1)):X2=ASC(RIGHT$(A$,1))
200 IFX1>57THENX1= X1-55:GOTO220
210 X1= X1-48
220 IFX2>57THENX2= X2-55:GOTO240
230 X2= X2-48
240 POKE AD,X1*16+X2:AD=AD+1:GOTO180
250 :
260 DATA A9,FF,8D,00,04,A9,0C,8D
270 DATA 00,D8,60,              -1
READY.
```

Figura 1. Rappresenta l'adattamento per il Commodore 64 dello schema di figura 2 pubblicata sul n. 6 (pag. 47) di Commodore Computer Club.

```
100 PRINT"SYS 826 FA APPARIRE IN ALTO"
110 PRINT"A SINISTRA SULLO SCHERMO DEL"
120 PRINT"VIC 20 INESPANSO UN CARATTERE (X)"
130 PRINT"SELEZIONABILE CON POKE 827,X."
140 PRINT"COLORE (Y) SELEZIONABILE CON:"
150 PRINT"POKE 832,Y"
160 AD=826:AD$="SYS 826"
170 :
180 READ A$:IFVAL(A$)<0THENPRINT""AD$:END
190 X1=ASC(LEFT$(A$,1)):X2=ASC(RIGHT$(A$,1))
200 IFX1>57THENX1= X1-55:GOTO220
210 X1= X1-48
220 IFX2>57THENX2= X2-55:GOTO240
230 X2= X2-48
240 POKE AD,X1*16+X2:AD=AD+1:GOTO180
250 :
260 DATA A9,FF,8D,00,1E,A9,0C,8D
270 DATA 00,96,60,              -1
READY.
```

Figura 1/A. Idem come figura 1 per Vic 20 inespanso.

```
100 PFINT"SYS 826 RIEMPE LO SCHERMO DEL"
110 PRINT"COMMODORE 64 ISTANTANEAMENTE
120 PRINT"RIPETENDO UN CARATTERE (X)"
130 PRINT"SELEZIONABILE CON POKE 847,X"
140 PRINT"COLORE (Y) SELEZIONABILE CON:"
150 PRINT"POKE 852,Y"
160 AD=826:AD$="SYS 826"
170 :
180 READ A$:IFVAL(A$)<0THENPRINT""AD$:END
190 X1=ASC(LEFT$(A$,1)):X2=ASC(RIGHT$(A$,1))
200 IFX1>57THENX1= X1-55:GOTO220
210 X1= X1-48
220 IFX2>57THENX2= X2-55:GOTO240
230 X2= X2-48
240 POKE AD,X1*16+X2:AD=AD+1:GOTO180
250 :
260 DATA A9,04,8D,52,03,A9,00,8D
270 DATA 51,03,A9,00,8D,56,03,A9
280 DATA D8,8D,57,03,A9,FF,8D,00
290 DATA 07,A9,0C,8D,00,D8,EE,51
300 DATA 03,EE,56,03,D0,EE,EE,52
310 DATA 03,EE,57,03,AD,52,03,C9
320 DATA 08,D0,E1,60,          -1

READY.
```

Figura 2. Adattamento per Commodore 64 dello schema di figura 3 dello stesso numero.

```
100 PRINT"SYS 826 RIEMPE LO SCHERMO DEL"
110 PRINT"VIC 20 INESPANSO ISTANTANEAMENTE
120 PRINT"RIPETENDO UN CARATTERE (X)"
130 PRINT"SELEZIONABILE CON POKE 847,X"
140 PRINT"COLORE (Y) SELEZIONABILE CON:"
150 PRINT"POKE 852,Y"
160 AD=826:AD$="SYS 826"
170 :
180 READ A$:IFVAL(A$)<0THENPRINT""AD$:END
190 X1=ASC(LEFT$(A$,1)):X2=ASC(RIGHT$(A$,1))
200 IFX1>57THENX1= X1-55:GOTO220
210 X1= X1-48
220 IFX2>57THENX2= X2-55:GOTO240
230 X2= X2-48
240 POKE AD,X1*16+X2:AD=AD+1:GOTO180
250 :
260 DATA A9,1E,8D,52,03,A9,00,8D
270 DATA 51,03,A9,00,8D,56,03,A9
280 DATA 96,8D,57,03,A9,FF,8D,00
290 DATA 1E,A9,0C,8D,00,96,EE,51
300 DATA 03,EE,56,03,D0,EE,EE,52
310 DATA 03,EE,57,03,AD,52,03,C9
320 DATA 20,D0,E1,60,          -1

READY.
```

Figura 2/A. Idem come figura 2 per Vic 20 inespanso.

```
100 PRINT"SYS 826 FA APPARIRE NELLA"
110 PRINT"PRIMA RIGA DI SCHERMO DEL 64"
120 PRINT"UN MESSAGGIO IL CUI CONTENUTO"
130 PRINT"E' LUNGO 40 CARATTERI ALLOCATI"
140 PRINT"DALLA LOCAZIONE 845 IN POI."
150 PRINT"COLORE (X) SELEZIONABILE CON:"
160 PRINT"POKE 835,X"
170 :
180 AD=826:AD$="SYS 826"
190 :
200 READ A$:IFVAL(A$)<0THEN PRINT AD$:END
210 X1=ASC(LEFT$(A$,1)):X2=ASC(RIGHT$(A$,1))
220 IFX1>57THENX1= X1-55:GOTO240
230 X1= X1-48
240 IFX2>57THENX2= X2-55:GOTO260
250 X2= X2-48
260 POKE AD,X1*16+X2:AD=AD+1:GOTO200
270 :
280 DATA A2,00,BD,4D,03,9D,00,04
290 DATA A9,0C,9D,00,D8,E8,E0,28
300 DATA D0,F0,60,2A,2A,2A,2A,2A
310 DATA 2A,2A,20,03,0F,0D,0D,0F
320 DATA 04,0F,12,05,20,03,0F,0D
330 DATA 10,15,14,05,12,20,03,0C
340 DATA 15,02,20,2A,2A,2A,2A,2A
350 DATA 2A,2A,2A,          -1
READY.
```

Figura 3. La ... sorpresa di questa puntata.

```
100 PRINT"SYS 826 FA APPARIRE NELLA"
110 PRINT"PRIMA RIGA DI SCHERMO DEL VIC 20 INESPANSO"
120 PRINT"UN MESSAGGIO IL CUI CONTENUTO"
130 PRINT"E' LUNGO 40 CARATTERI ALLOCATI"
140 PRINT"DALLA LOCAZIONE 845 IN POI."
150 PRINT"COLORE (X) SELEZIONABILE CON:"
160 PRINT"POKE 835,X"
170 :
180 AD=826:AD$="SYS 826"
190 :
200 READ A$:IFVAL(A$)<0THEN PRINT AD$:END
210 X1=ASC(LEFT$(A$,1)):X2=ASC(RIGHT$(A$,1))
220 IFX1>57THENX1= X1-55:GOTO240
230 X1= X1-48
240 IFX2>57THENX2= X2-55:GOTO260
250 X2= X2-48
260 POKE AD,X1*16+X2:AD=AD+1:GOTO200
270 :
280 DATA A2,00,BD,4D,03,9D,00,1E
290 DATA A9,0C,9D,00,96,E8,E0,16
300 DATA D0,F0,60,03,0F,0D,0D,0F
320 DATA 04,0F,12,05,20,03,0F,0D
330 DATA 10,15,14,05,12,20,03,0C
340 DATA 15,02,20,          -1
READY.
```

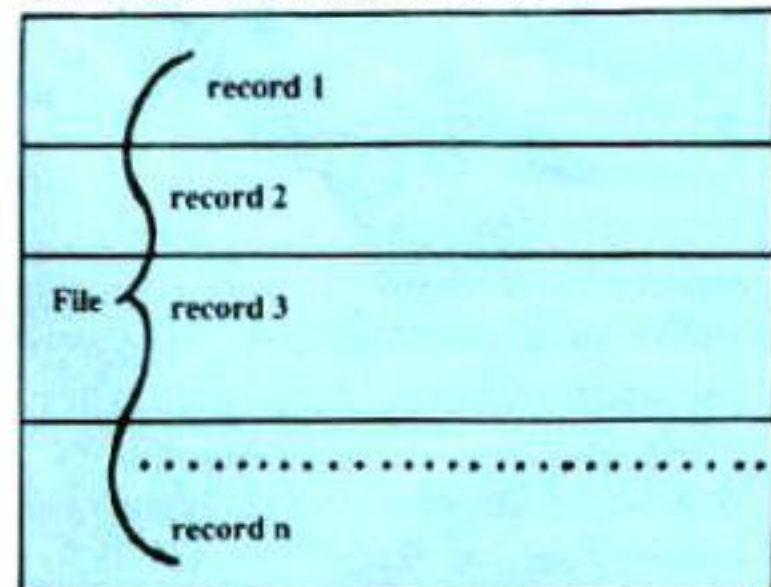
Figura 3/A. Come figura 3 per il Vic 20.

FILE RELATIVI

L'intenzione di questo articolo è quella di rendere chiaro a chi, o vuoi per mancanza di letteratura specifica o vuoi per la stringatezza dei manuali che solitamente accompagnano il CBM 64 e periferiche annesse, non ha avuto modo di potersi avvicinare alla strutturazione dei file relativi, trattati dal DOS del CBM 64 e alla programmazione di routine che implementino le istruzioni inerenti alla gestione dei file (o archivi) relativi.

Iniziamo innanzitutto col dire che un file è una struttura di informazioni di natura alfanumerica immagazzinata nella memoria di massa di un elaboratore, (generalmente supporti magnetici come ad esempio, nastri o dischi). Il singolo elemento del file viene denominato record, a sua volta il record può essere diviso in più parti elementari chiamati campi, (se non avete ancora le idee confuse, il singolo campo può essere diviso in sottocampi).

Ricapitolando, il nostro file è un aggregato di record, il record è un aggregato di campi, pertanto, se consideriamo il nostro file come un insieme, i record sono i sottoinsiemi appartenenti al file ed i campi i singoli elementi del record.

	record 1
	record 2
File	record 3
	....
	record n

campo 1	campo 2	...	campo n	Record

Facciamo un esempio concreto per consolidare ciò che abbiamo appreso sino ad ora.

Consideriamo l'esempio ormai classico della rubrica telefonica. Il file sarà quindi la rubrica ed esso conterrà le informazioni che possono essere: cognome, nome, indirizzo, CAP, città e telefono. Il record sarà così composto:

Record

cognome	nome	indirizzo	Cap	città	telefono
campo 1	campo 2				...campo 6

Passiamo ora a descrivere come tali archivi vengono memorizzati sui dischetti magnetici. Oramai tutti sanno che un dischetto è composto da tante tracce concentriche (35, nel caso dei dischetti utilizzati dal CBM 64) ed ogni traccia è suddivisa in settori. Consideriamo una traccia rettificata:

blocco1	Blocco 2	...	blocco n

Ogni settore prende il nome di blocco e viene individuato da 2 parametri. Numero della traccia e numero del settore. Un blocco è composto da 256 byte ma usando gli archivi relativi, un singolo blocco può contenere al massimo 254 byte di informazioni, gli altri 2 servono al DOS per conoscere l'indirizzo del prossimo blocco dati (traccia e settore). Vediamo ora di capire che differenza sussiste fra il record logico e il record fisico. Possiamo considerare come record logico il record pocanzi visto, costituito da cognome, nome, ecc., e, come esso, ogni altro tipo di record che più ci aggrada. In definitiva il record logico viene creato dal programmatore. Si definisce record fisico la più piccola quantità di informazioni che può essere passata dalla periferica all'elaboratore o viceversa.

Nel nostro caso il record fisico è proprio il famigerato blocco. Il DOS, infatti, scrive nel BUFFER di I/O l'intero contenuto del blocco: saremo noi a prelevare la quantità di dati che interessa. Cosicché in un blocco si potranno scrivere più record logici od un record logico avente lunghezza pari al record fisico stesso. A questo punto è bene capire come mai il massimo di numeri caratteri che può contenere un singolo record logico di un file relativo sia 254, (vedremo in seguito come superare questa limitazione). Poco fa abbiamo accennato ai BUFFER di I/O. Essi non sono altro che una zona di memoria RAM, interna al 1541, di lunghezza pari a 256 byte ognuno. Servono per memorizzare temporaneamente i dati che transitano dall'elaboratore al drive e viceversa (l'istruzione OPEN fa sì che venga assegnato un BUFFER di I/O al file aperto).

COMMODORE SCHOOL

Per creare un file relativo occorre come prima operazione, definire la lunghezza del record (logico); supponiamo di avere la nostra agenda telefonica con i seguenti campi:

cognome	caratteri := 20 (campo 1)
nome	caratteri := 10 (campo 2)
indirizzo	caratteri := 25 (campo 3)
C.A.P.	caratteri := 5 (campo 4)
città	caratteri := 20 (campo 5)
telefono	caratteri := 12 (campo 6)
TOTALE	caratteri : = 92 (record)

Ogni campo dovrà essere fisicamente separato dall'altro interponendo fra di essi un CARRIAGE RETURN (chr$(13)), quindi, nel conteggio totale dei caratteri del record dovranno essere sommati i chr$(13), uno per campo. Il nostro record sarà lungo 92+6=98 caratteri. Esaminiamo adesso la sintassi del comando di apertura di un file relativo:

OPEN FL,DI,CL,"NOME FILE,L,"+CHR$(LUNGH.REC.)
FL := numero del file logico (da 1 a 127);
DI := numero del dispositivo, normalmente 8;
CL := numero del canale, (può assumere i valori da 2 a 14);
NOME FILE := è ovviamente il nome del file;
,L, := indica che il file è di tipo relativo;
CHR$(LUN) := è la lunghezza in caratteri del record, (compresi i chr$(13)).

Dal momento che viene eseguito il comando OPEN, il DOS annoterà nella directory il nome del file specificando che è di tipo relativo, la traccia ed il settore di inizio del primo blocco di dati, la lunghezza del record, l'indirizzo del primo SIDE SECTOR (vedremo dopo cos'è) ed altre cosucce che per ora non ci riguardano.
Una volta che il file è organizzato, per poterci scrivere o leggere si useranno le conosciute istruzioni INPUT#, PRINT# e GET#. Per aprire ul file relativo già esistente la sintassi del comando OPEN è ancora più semplice:

OPEN FL,DI,CL,"NOME FILE"

A differenza dei file sequenziali, in un file relativo, una volta aperto, vi ci si può indifferentemente leggere o scrivere ed in qualsiasi suo punto. Facciamo ora conoscenza con i SIDE SECTOR:

Sappiamo che un file è un aggregato di record logici. Ogni record interno al file può essere individuato da un proprio numero d'ordine o indice posizionale. Ciascun record logico verrà scritto in blocco di memoria di massa, ma il blocco è individuato da 2 parametri. Sarà quindi il DOS a stabilire la corrispondenza, (biunivoca), tramite i SIDE SECTOR, fra il numero del record e relativo indirizzo fisico di memorizzazione. In un blocco possono entrarci anche più di un record logico (ricordate?) quindi il DOS terrà conto anche di record spezzati su 2 blocchi.

I SIDE SECTOR sono fisicamente dei blocchi di memoria di massa che contengono sia l'indirizzo di altri SIDE SECTOR che l'indirizzo di inizio di 120 blocchi di dati. Ciò vuol dire che associati ad un file con più di 120 blocchi ci sono più di 1 SIDE SECTOR e possono esservene fino a 6, (che equivalgono a 720 blocchi di dati. E' chiaro che se ne potranno utilizzare solamente 658, 664 blocchi liberi - 6 SIDE SECTOR). Per passare al DOS il numero del record esiste una istruzione apposita:

PRINT#FL,"P"CHR$(CANALE)CHR$(LO)CHR$(HI)CHR$(PUNTATORE)

Questo comando va inviato al canale comandi pertanto occorre prima scrivere OPEN FL,8,15. Dato che i record possono essere più di 256 non basta un byte per individuare un record, occorrono quindi 2 byte ed è per questo che nella sintassi del comando troviamo CHR$(LO) e CHR$(HI). Valgono le seguenti formulette:

HI := INT(nr.RECORD/256);
LO := nr.RECORD - HI*256;
CHR$(PUNTATORE) punta al carattere del record.

Mettiamo in pratica quanto è stato detto sino ad ora.
Approfittando dell'occasione vediamo di crearci qualcosa di utile. Rubriche telefoniche ce ne sono troppe e un data base è un pochettino difficile di primo acchito. Potremmo costruire un archivio che memorizzi le prove HARDWARE apparse sulle riviste italiane, così evitiamo di mettere ogni volta a soqquadro la nostra libreria nel tentativo di riuscire a trovare quanto cercato. Il nostro record avrà dunque il seguente tracciato:

titolo articolo	caratteri := 30
nome rivista	caratteri := 20
numero rivista	caratteri := 3
TOTALE	caratteri := 53
numero CHR$(13)	:= 3
TOTALE	caratteri := 56

Morale: abbiamo creato il nostro file, ed il DOS ha predisposto in memoria lo spazio sufficiente per 256 record (linea 250). In seguito, questo spazio potrà anche essere espanso. (**Figura 2**)

Bene: abbiamo concluso il programma, esso chiaramente ha moltissimi limiti.
Il primo è quello che bisogna ricordarsi il numero del record prima di cercarlo ma, credo che riuscirete da soli a creare una chiave di ricerca con 2 vettori, 1 per la chiave ed 1 per il numero del record corrispondente.

N.B. L'istruzione "PRINT#"non può essere abbreviata con "?".

```
100 REM **** UTILIZZO DEL FILE RELATIVE ****
110 REM ****      DI NOME HARD - TEST     ****
120 :
130 PRINT"INSERISCI IL DISCO CONTENENTE
140 PRINT"IL FILE HARD-TEST E PREMI UN TASTO"
150 GET X$: IF X$="" THEN 150
160 OPEN 2,8,2,"HARD-TEST"
170 DEF FN A(X)=INT(X/256): REM NUMERO RECORD HI
180 DEF FN B(Y)=Y-HI*256  : REM NUMERO RECORD LO
190 OPEN 1,8,15
200 :
210 REM MENU PRINCIPALE
220 :
230 PRINT"                 MENU"
240 PRINT"  [1] - FINE LAVORO"
250 PRINT"  [2] - LETTURA FILE"
260 PRINT"  [3] - SCRITTURA FILE"
270 PRINT"           SCEGLI ";
280 INPUTA:IF A>3 OR A<1 THEN 280
290 ON A GOTO 320,360,490
300 :
310 REM CHIUSURA LAVORO
320 CLOSE 2: CLOSE 1: END
330 :
340 REM LETTURA RECORD
350 :
360 PRINT"  ":INPUT"NUMERO DEL RECORD :=";NR
370 IF NR >256 OR NR<1 THEN 360
380 HI=FN A(NR):LO=FN B(NR)
390 PRINT#1,"P"CHR$(2)CHR$(LO)CHR$(HI)CHR$(1):GOSUB650
400 INPUT#2,F1$,F2$,F3$:GOSUB650
410 PRINT"NOME ARTICOLO ---> ";F1$
420 PRINT"NOME RIVISTA  ---> ";F2$
430 PRINT"NUM. RIVISTA  ---> ";F3$
440 INPUT"PREMI <RETURN>";AA
450 GOTO 230
460 :
470 REM SCRITTURA RECORD
480 :
490 INPUT"NUMERO DEL RECORD :=";NR
500 HI=FN A(NR):LO=FN B(NR)
510 PRINT#1,"P"CHR$(2)CHR$(LO)CHR$(HI)CHR$(1):GOSUB650
520 INPUT"NOME ARTICOLO ---> ";F1$
530 IF LEN (F1$)>30 THEN 520
540 INPUT"NOME RIVISTA  ---> ";F2$
550 IF LEN (F2$)>20 THEN 540
560 INPUT"NUM. RIVISTA  ---> ";F3$
570 IF LEN (F3$)>3 THEN 560
580 INPUT"VUOI CORREGGERE (S/N)";Y$
590 IF Y$="S" THEN520
```

```
600 PRINT#2,F1$;CHR$(13);F2$;CHR$(13);F3$:GOSUB650
610 GOTO 230
620 :
630 REM CONTROLLO ERRORE DI INPUT DA DISCO
640 :
650 INPUT#1,A,A$,B,C
660 IF A=0 OR A=50 THEN RETURN
670 PRINT A,A$
680 PRINT"█FINE PROGRAMMA PER ERRORE"
690 CLOSE 2: CLOSE 1

READY.
```

```
100 REM *****      FILE RELATIVI      *****
110 :
120 REM ***** CREAZIONE DI UNO SPAZIO****
130 REM ***** SU DISCO ALLO SCOPO DI ****
140 REM *****   OSPITARE IN SEGUITO   ****
150 REM ***** 256 RECORD LUNGHI 56 B.****
160 :
170 :
180 REM CREAZIONE DI UNA STRINGA - CODICE
190 FOR I=1 TO 55: CA$=CA$+".": NEXT
200 DEF FNA(X)=INT(X/256): REM CALCOLO DI HI
210 DEF FNB(Y)=Y-HI*256  : REM CALCOLO DI LO
220 OPEN 1,8,15:REM APERTURA CANALE COMANDI
230 REM LUNGHEZZA RECORD =56 COMPRESI CHR$(13)
240 OPEN 2,8,2,"HARD-TEST,L,"+CHR$(56)
250 FOR I=256 TO 1 STEP-1: HI=FNA(I): LO=FNB(I)
260 PRINT#1,"P"CHR$(2)CHR$(LO)CHR$(HI)CHR$(1)
270 PRINT#2, CA$: GOSUB 300
280 PRINT "RECORD N." I: NEXT
290 CLOSE 1: CLOSE 2: END
300 INPUT#1,A,A$,B,C
310 IF A=0 OR A=50 THEN RETURN
320 PRINT A,A$
330 PRINT"█IL PROGRAMMA TERMINA IN ERRORE":END

READY.
```

IL GIOCO DEL 15

Simulazione del solitario dei vecchi tempi.

Sul video appare, se avete trascritto tutto correttamente, una griglia di 4 x 4 caselle, quindici delle quali contengono, appunto, casualmente i numeri da 1 a 15. Il gioco consiste nello spostare le tessere fino a che non sono ordinate come nei REM delle righe da 170 a 230.

E' sufficiente indicare il valore della tessera che si desidera spostare nel posto vuoto.

Vito Ascoli - Napoli

```
100 REM ***********************************
110 REM ***  GIOCO DEL 15   PER   C 64   ***
120 REM ***      DI   ASCOLI   VITO      ***
130 REM ***     VIA ENRICO RUTA 1        ***
140 REM ***  NAPOLI   TEL. 081/412507    ***
150 REM ***********************************
160 :
170 REM E' NECESSARIO ORDINARE LE
180 REM CASELLE NEL MODO SEGUENTE:
190 :
200 REM         1  5   9  13
210 REM         2  6  10  14
220 REM         3  7  11  15
```

```
230 REM        4   8   12
240 :
250 PRINTCHR$(8):DIMN(4,4)
260 X$(1)="■■■■■■■■■■■■■":X$(2)="□■■■■■■■■■■■■■■■■■"
270 X$(3)="□■■■■■■■■■■■■■■■■■■■■■■■■":X$(4)="□■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■"
280 :
290 :
300 REM ** STAMPA DEL QUADRO DI GIOCO **
310 PRINT"■□■          ─────────────────────────────────────────"
320 PRINT"      ■──────── GIOCO DEL QUINDICI ────────"
330 PRINT"      ─────────────────────────────────────────────■■"
340 PRINTSPC(11)"┌─────┬─────┬─────┬─────┐"
350 FORI=1TO4:PRINTSPC(11)
360 FORJ=1TO4:PRINT"|     ";:NEXT
370 PRINT"|"
380 PRINTSPC(11)"├─────┼─────┼─────┼─────┤"
390 NEXT
400 PRINTSPC(11)"□└─────┴─────┴─────┴─────┘"
410 MOSSE=1
420 REM: PROVA. K=1:FORI=1TO4:FORJ=1TO4:N(J,I)=K
430 REM  N(J,I)=K:K=K+1:NEXTJ,I:N(4,4)=0
440 :
450 REM ** INIZIALIZZAZIONE CASUALE **
460 FORK=1TO15
470 I=INT(RND(1)*4+1):J=INT(RND(1)*4+1)
480 IFN(I,J)<>0THEN470
490 N(I,J)=K
500 NEXTK
510 :
520 GOSUB1060:REM POSIZIONAMENTO NUMERI
530 :
540 REM  ** RICHIESTA N-MA MOSSA **
550 PRINT"■■■■          ┌─────────────────────────────────┐"
560 PRINT"          | BATTI LA TUA MOSSA N.";MOSSE:PRINT"□"SPC(33)"|"
570 PRINT"          └─────────────────────────────────┘"
580 M$="":INPUT"                              ";M$:M=VAL(M$)
590 REM  ** SOLO NUMERI TRA 1 E 15 ***
600 IFM>0ANDM<16THEN680
610 PRINT"□                              "
620 PRINT"□□□"SPC(8)"■ SOLO NUMERI DA 1 A 15 ■ "
630 FORT=1TO1000:NEXT
640 PRINT"□□□□□□□□":GOTO550
650 :
660 :
670 REM    ** ALGORITMO CENTRALE **
680 FORI=1TO4:FORJ=1TO4:IFN(I,J)=MTHEN700
690 NEXTJ,I
700 II=I+1:JJ=J:GOSUB970:IFF=1THEN820
710 II=I-1:JJ=J:GOSUB970:IFF=1THEN820
720 II=I:JJ=J+1:GOSUB970:IFF=1THEN820
730 II=I:JJ=J-1:GOSUB970:IFF=1THEN820
```

```
740 :
750 REM    ** MOSSA NON LECITA **
760 PRINT"□                            "
770 PRINT"□□□"SPC(8)"  ▘ MOSSA NON LECITA ■     "
780 FORT=1TO2000:NEXT
790 PRINT"□□□□□□□□":GOTO550
800 :
810 REM** ESECUZIONE MOSSA RICHIESTA **
820 N(I,J)=0:N(II,JJ)=M:MOSSE=MOSSE+1
830 IFN(3,4)=15ANDN(2,4)=14ANDN(1,4)=13THEN850
840 GOTO910
850 IFN(4,3)=12ANDN(3,3)=11ANDN(2,3)=10ANDN(1,3)=9THEN870
860 GOTO910
870 IFN(4,2)=8ANDN(3,2)=7ANDN(2,2)=6ANDN(1,2)=5THEN890
880 GOTO910
890 IFN(4,1)=4ANDN(3,1)=3ANDN(2,1)=2ANDN(1,1)=1THEN1160
900 REM    *********      VITTORIA ***********
910 PRINT"□":GOSUB1060
920 PRINT"■■■■■■■■"SPC(11)"                  □□□□□□□□□":GOTO550
930 :
940 :
950 REM ** CONTROLLO LICEITA' MOSSA **
960 REM ** (SE F=1 MOSSA CONSENTITA) **
970 F=0
980 IFII<1ORII>4ORJJ<1ORJJ>4THEN1010
990 IFN(II,JJ)<>0THEN1010
1000 F=1
1010 RETURN
1020 :
1030 :
1040 REM ** POSIZIONAMENTO DEI NUMERI **
1050 REM ** NELLE RISPETTIVE CASELLE  **
1060 PRINT"■■■■■":FORI=1TO4:PRINT
1070 FORJ=1TO4
1080 IFN(I,J)=0THENPRINTX$(J)"    ":GOTO1100
1090 PRINTX$(J);N(I,J)
1100 NEXTJ
1110 NEXTI
1120 RETURN
1130 :
1140 :
1150 REM    ** FINE DEL GIOCO **
1160 GOSUB1060
1170 PRINT"■■■■■          | ▘ HAI VINTO IN";MOSSE"■ MOSSE ■"
1180 PRINT"■■         VUOI GIOCARE ANCORA? (S/N)"
1190 GETA$:IFA$=""THEN1190
1200 IFA$="N"THENPRINT"□":END
1210 IFA$<>"S"THEN1190
1220 FORI=1TO4:FORJ=1TO4:N(I,J)=0:NEXTJ:NEXTI:GOTO310

READY.
```

SPACE DEFENDER

```
100 REM ** SPACE-DEFENDER PER CBM 64   **
110 REM ** JOHN CERESI , VIA MAZZINI   **
120 REM ** 18016 S.BARTOLOMEO AL MARE **
130 REM ** TEL. 0183/400030 **
140 :
150 TS=0:O=54272:T=0:JO=0
160 DEFFNR(X)=1144+40*(INT(RND(0)*20)):GOTO700
170 PRINT"■":POKE53281,12:N=1464:F=55736:SC=0:D=5
180 PRINT"■■■          ■":PRINT"■■PUNTEGGIO = ";SC
190 POKEN,90
200 I=FNR(X):J=FNR(X):K=FNR(X):H=FNR(X)
210 IFH=IORH=JORH=KORI=JORJ=KTHEN200
220 POKEH,60:POKEI,42:POKEJ,60:POKEK,42
230 REM **INPUT TASTIERA**
240 IFJO=100THEN1100
250 GETA$:IFA$="■"THEN800
260 IFA$="↑"THENPOKEN,32:N=N-40:F=F-40:GOTO530
270 IFA$="■"THENPOKEN,32:N=N+40:F=F+40:GOTO530
280 L=L+1:IFL<DTHEN240
290 H=H-1:I=I-1:J=J-1:K=K-1:L=0
300 IF(H-1024)/40=INT((H=1024)/40)THEN560
310 IF(I-1024)/40=INT((I-1024)/40)THEN560
320 IF(J+1024)/40=INT((J-1024)/40)THEN560
330 IF(K-1024)/40=INT((K-1024)/40)THEN560
340 POKEH+1,32:POKEI+1,32:POKEJ+1,32:POKEK+1,32
350 POKEH,60:POKEI,60:POKEJ,60
360 POKEK,60:POKEH+O,2:POKEI+O,2:POKEJ+O,2:POKEK+O,2
370 GOTO240
380 REM ** CONTOLLO SE HO COLPITO IL NEMICO **
390 FORM=200TO220:POKEN+O,1:POKEN+O,2:NEXT
```

```
400 IFH>NANDH<N+40THEN450
410 IFI>NANDI<N+40THEN460
420 IFJ>NANDJ<N+40THEN470
430 IFK>NANDK<N+40THEN480
440 SC=SC-5:GOTO500
450 B=H:H=FNR(X):GOTO950
460 B=I:I=FNR(X):GOTO950
470 B=J:J=FNR(X):GOTO950
480 B=K:K=FNR(X):GOTO950
490 SC=SC+10
```

```
500 PRINT"■■■       ■":PRINT"■■PUNTEGGIO = ";SC
510 IFSC>500THEND=0
520 GOTO240
530 IFN<1104THENN=1104
540 IFN>1984THENN=1984
```

```
550 POKEN,90:POKEN+O,0:GOTO240
560 FORC=55377TO56275STEP40:POKEC,2:NEXTC:FORC=1TO100:NEXTC
570 FORC=55377TO55275STEP40:POKEC,1:NEXTC
580 REM **FINE GIOCO**
590 PRINT"FINE GIOCO"
600 FORC=1TO1000:NEXTC:POKE53281,6:POKE53280,14
610 PRINT"PUNTEGGIO FINALE = ";SC:PRINT""
620 IFSC>TSTHENTS=SC
630 PRINT"PUNTEGGIO MAX = ";TS
640 PRINT"GIOCHI ANCORA  (Y/N)"
650 GETA$:IFA$=""THEN650
660 IFA$="Y"THEN170
670 IFA$="N"THEN890
680 GOTO650
690 REM **INIZIO**
700 POKE53281,12:PRINT"":PRINTTAB(8);"SPACE-DEFENDER"
710 PRINTTAB(13)"BY JOHN"
720 PRINT"PREMI ↑ PER ANDARE SU":PRINT"PREMI CRSR GIU' PER ANDARE GI
730 PRINT"PREMI F7 ":PRINT"PER SPARARE"
740 PRINT"TIENI LONTANO GLI INVASORI"
750 PRINT"DAL TUO SPAZIO"
760 PRINT"PIGIA UN TASTO PER INIZIARE"
770 IFPEEK(197)=64THEN770
```

```
780 GOTO1040
790 REM **SPARO LASER**
800 POKE54296,25
810 POKE54277,20
820 POKE54278,175
830 POKE54276,33
840 FORZ=1TO7:POKE54273,RND(1)*256:NEXT
850 POKE54276,32
860 POKE54296,0
870 GOTO380
880 REM **FINE GIOCO**
890 PRINTTAB(14)"CIAO"
900 PRINT:FORP=1TO50:NEXT:T=T+1
910 IFT=20THEN930
920 GOTO900
930 PRINT"":END
940 REM **SCIA LASER**
950 FORW=1TO(B-N)
960 POKEF+W,0
970 POKEN+W,45:POKEN+W-1,32
980 POKEN,90
990 NEXTW
1000 POKEB,102
1010 POKEN+40,32
1020 POKEB,32
1030 GOTO490
```

```
1040 PRINT"CON CHE COSA PREFERISCI GIOCARE :"
1050 PRINT"JOYSTIK (1) O TASTIERA (J O T) ?"
1060 GETA$:IFA$=""THEN1060
1070 IFA$="J"THENJO=100:GOTO170
1080 IFA$="T"THEN170
1090 GOTO1060
1100 SS=NOTPEEK(56321)AND15:FF=PEEK(56321)AND16
1110 IF(SSAND1)THENPOKEN,32:N=N-40:F=F-40:GOTO530
1120 IF(SSAND2)THENPOKEN,32:N=N+40:F=F+40:GOTO530
1130 IFFF=0THEN800
1140 GOTO280
READY.
```

LABIRINTO

Un simpatico gioco basato sulla ridefinizione di alcuni caratteri.

Il programma gira sulla configurazione del Vic 20 inespanso ed occupa tutta la memoria; quindi, durante la copiatura, si consiglia di tralasciare le REM all'inizio del listato in modo da evitare messaggi del tipo OUT OF MEMORY.

Il gioco consiste nel riuscire ad aprirsi un varco in mezzo ad un labirinto, formato da caratteri a forma di blocco di cemento, in modo da raggiungere una delle bandierine poste sui quattro lati dello schermo entro un minuto di tempo. Per aiutarsi, si hanno a disposizione tre proiettili con cui è possibile abbattere dei blocchi e liberarsi così un passaggio.
Anche alla fine di ogni labirinto, al proprio punteggio si aggiungono tanti punti quanti sono i secondi avanzati; inoltre il livello di difficoltà aumenta ad ogni labirinto, stabilizzandosi solo dal sesto labirinto in poi.

Durante il gioco, nella parte superiore dello schermo compaiono varie informazioni: tempo, punteggio, numero dei labirinti e numero dei proiettili a disposizione.

Per muoversi si adoperano i tasti: ",," (rotazione a sinistra); "." (rotazione a destra); "Z" (per avanzare di un passo) e "*" per sparare.

Ed ora, buon divertimento...!

Pietro Pino
Via G. Mameli, 3
20129 Milano

```
15 REM LABIRINTO VIC 20 INESPANSO
20 REM BY PIETRO PINO
30 REM VIA MAMELI 3-MILANO
40 REM TEL. 735763
95 GOTO9000
97 DIMM(4)
100 :
110 POKE36879,8:POKE36878,15:POKE36869,254
115 LE=LE+1:SP=3:M(1)=-22:M(2)=1:M(3)=22:M(4)=-1
120 FL=0:PRINT"■■";
130 PRINT"■LEVEL:■                ■SCORE:        ■ 000000  ■"SC
131 PRINT"■SHOTS:■                ■ ***   ";
133 IFLE>6THENLE=6
135 FORK=1TOLE:POKE7680+21+K,158:NEXT
140 S=7680+88
150 FORK=STOS+417
160 IFRND(1)*100>(60-LE*4)THENPOKEK,0
165 NEXT
167 GOTO2000
170 CH=1:TI$="000000"
180 P=7954
185 POKEP,CH
190 PRINT"■■"SPC(8)"■"TI$
195 IFTI$>"000059"THENPRINT"■■GAME OVER":END
200 T=PEEK(197)
```

```
210 IFT=29THENCH=CH-1:POKE36876,200:IFCH<1THENCH=4
215 IFT=37THENCH=CH+1:POKE36876,210:IFCH>4THENCH=1
220 IFT=33THEN500
230 IFT=14ANDFL=0THENGOTO3010
231 POKE36876,0
233 FORT=1TO80:NEXT
235 POKE36876,0:GOTO185
500 REM MOVIMENTO
505 P1=P
520 P=P+M(CH):GOTO527
527 IFPEEK(P)=5THEN6000
530 IFP>S+417ORP<SORPEEK(P)<>32THENP=P-M(CH):GOTO185
540 FORT=1TO50:NEXT:POKEP1,32:POKE36876,190:GOTO235
2000 :
2010 POKES+RND(1)*22,5
2020 POKES+(INT(RND(1)*18))*22,5
2025 POKES+(INT(RND(1)*18))*22+21,5
2030 POKES+18*22+RND(1)*22,5
2040 GOTO170
3010 R=M(CH):P1=P
3020 IFPEEK(P1+R)=0THEN3040
3023 IFP1+R<SORP1+R>S+417THEN3053
3025 POKEP1+R,42:FORT=1TO70:NEXT:POKEP1+R,32:
3030 P1=P1+R:GOTO3020
3040 P1=P1+R
3043 POKE36877,200
3045 FORL=1TO15:POKEP1,102:FORT=1TO10:NEXT:POKEP1,32
3046 FORT=1TO10:NEXT:POKE36878,15-L:NEXT
3050 POKE36877,0:POKE36878,15:
3053 SP=SP-1:IFSP=0THENPOKE7741,160:FL=1:GOTO190
3055 FORK=1TOSP:POKE7740+K,42+128:NEXT:POKE7740+K,160:GOTO190
6000 T$=RIGHT$(TI$,2):PRINT""TAB(8)"< "T$" >"
6005 PRINT""TAB(8)"SECONDI"
6007 PRINT""TAB(8)"BONUS:"
6010 PRINT""TAB(6)"60-"T$" ="60-VAL(T$)
6020 PRINT""TAB(3)"┌──────────────────┐"
6025 PRINTTAB(3)"│ TIENITI PRONTO │"
6030 PRINTTAB(3)"└──────────────────┘"
6040 SC=SC+60-VAL(T$)
6050 FORT=1TO5000:NEXT:GOTO100
9000 POKE56,24:POKE52,24:FORK=0TO192*8-1
9005 POKE6144+K,PEEK(32768+K):NEXT
9010 :
9020 READQ
9025 IFQ=-1THEN9040
9030 POKE6144+W,Q:W=W+1:GOTO9020
9040 POKE36869,254
9045 GOTO100
9050 DATA63,67,255,135,135,135,134,252
9055 DATA8,28,8,8,93,93,127,65
9060 DATA0,240,64,114,127,114,64,240
9065 DATA65,127,93,93,8,8,28,8
9070 DATA0,15,2,78,254,78,2,15
9075 DATA124,60,28,60,124,4,4,4,-1
READY.
```

SINTETIZZATORE SONORO

Il programma trasforma il Commodore 64 in un perfetto sintetizzatore offrendo, inoltre, la possibilità di scegliere il suono più gradito. I tasti da usare riguardano la fila dei tasti che va dalla lettera «A» al simbolo « + », mentre nella fila superiore vi sono i diesis.

Volendo usare una scala con suoni più bassi di una ottava, si digita il tasto SHIFT insieme al tasto della nota. Il volume si regola con i tasti di funzione (F1, F2, F3, F4). Per creare un effetto di sustain, digitare il tasto « = ». Per eliminarlo, si usa il tasto di un registro qualsiasi.

Gianpaolo Calafiore
L.go L. Da Vinci, 1 - 20024 Moncalieri (To)

```
100 REM" C O M M O D O R E  6 4
110 REM" ------------------------
120 REM" GIANPAOLO CALAFIORE
130 REM"
140 REM"  TEL 011/605 10 98
150 REM"
160 REM"  --MOOGSOUND  64--
170 REM"
180 REM" ------------------------
190 PRINT"[CLR]":POKE53281,0:POKE53280,0
200 PRINTSPC(6);"****** MOOGSOUND   64 ******"
210 PRINT"________________________________________"
220 PRINT:PRINT:PRINT:GOTO810
230 S=54272
240 FORL=STOS+24:POKEL,0:NEXT
250 POKES+24,7
260 GETZ$
270 IFZ$="A"THENFH=17:FL=103:GOTO720
280 IFZ$="W"THENFH=16:FL=112:GOTO720
290 IFZ$="S"THENFH=13:FL=137:GOTO720
300 IFZ$="E"THENFH=20:FL=176:GOTO720
310 IFZ$="D"THENFH=21:FL=237:GOTO720
320 IFZ$="F"THENFH=23:FL=59:GOTO720
330 IFZ$="T"THENFH=24:FL=157:GOTO720
340 IFZ$="G"THENFH=26:FL=20:GOTO720
350 IFZ$="Y"THENFH=27:FL=160:GOTO720
360 IFZ$="H"THENFH=23:FL=69:GOTO720
370 IFZ$="U"THENFH=31:FL=3:GOTO720
380 IFZ$="J"THENFH=32:FL=219:GOTO720
390 IFZ$="K"THENFH=34:FL=207:GOTO720
400 IFZ$="O"THENFH=36:FL=225:GOTO720
410 IFZ$="L"THENFH=39:FL=18:GOTO720
420 IFZ$="P"THENFH=41:FL=101:GOTO720
```

```
430 IFZ$=":"THENFH=43:FL=219:GOTO720
440 IFZ$=";"THENFH=46:FL=116:GOTO720
450 IFZ$="*"THENFH=43:FL=58:GOTO720
460 IFZ$="="THENFH=52:FL=39:GOTO720
470 IFZ$="\"THENFH=16:FL=103:GOTO720
480 IFZ$="┌"THENFH=15:FL=129:GOTO720
490 IFZ$="¦"THENFH=14:FL=162:GOTO720
500 IFZ$=":"THENFH=13:FL=208:GOTO720
510 IFZ$="|"THENFH=13:FL=10:GOTO720
520 IFZ$="|"THENFH=12:FL=78:GOTO720
530 IFZ$="_"THENFH=11:FL=158:GOTO720
540 IFZ$="‾"THENFH=10:FL=247:GOTO720
550 IFZ$="‾"THENFH=10:FL=89:GOTO720
560 IFZ$="♥"THENFH=9:FL=196:GOTO720
570 IFZ$="○"THENFH=9:FL=56:GOTO720
580 IFZ$="♠"THENFH=8:FL=180:GOTO720
590 IFZ$="1"THENFH=0:FL=0:GOSUB900:GOTO770
600 IFZ$="2"THENFH=0:FL=0:GOSUB910:GOTO770
610 IFZ$="3"THENU=64:E2=33:UI=9:ML=0:GOTO780
620 IFZ$="4"THENU=65:E2=64:UI=1:ML=5:GOTO780
630 IFZ$="5"THENU=64:E2=17:UI=8:ML=0:GOTO790
640 IFZ$="6"THENU=64:E2=65:UI=9:ML=0:GOTO780
650 IFZ$="7"THENU=64:E2=65:UI=59:ML=0:GOTO780
660 IFZ$="+"THENML=20:GOTO790
670 IFZ$="▓"THENR1=0:GOTO760
680 IFZ$="▓"THENR1=3:GOTO760
690 IFZ$="▓"THENR1=7:GOTO760
700 IFZ$="▓"THENR1=15:GOTO760
710 GOTO260
720 POKES+1,FH:POKES,FL
730 POKES+4,U:IFE2=0THEN750
740 POKES+4,E2:IFUI=0THENRETURN
750 GOTO260
760 POKES+24,R1:GOTO260
770 POKES+5,OP:POKES+6,255:ML=1:GOTO260
780 POKES+3,8:POKES+2,0
790 POKES+5,UI:POKES+6,ML:IFUI=0THEN720
800 GOTO260
810 PRINTSPC(4);"FLAUTO       1","  VOLUME 0: F  1":PRINT
820 PRINTSPC(4);"ORGANETTO    2","  VOLUME 1: F  3":PRINT
830 PRINTSPC(4);"CLAVICEMBALO 3","  VOLUME 2: F  5":PRINT
840 PRINTSPC(4);"XILOFONO     4","  VOLUME 3: F  7":PRINT
850 PRINTSPC(4);"PIANO        5","  SUSTAIN      +":PRINT
860 PRINTSPC(4);"PIANO 2      6":PRINT
870 PRINTSPC(4);"CLARINETTO   7":PRINT
880 GOTO230
890 END
900 U=65:E2=64:UI=0:GOSUB780:U=17:E2=0:OP=255:RETURN
910 U=65:E2=64:UI=0:GOSUB780:U=33:E2=0:OP=88:RETURN

READY.
```

BOMBER

Un piccolo aereo si muove lungo lo schermo da sinistra a destra e lancia una bomba premendo il tasto "Z". E' necessario colpire i grattacieli alla loro sommità, altrimenti la bomba non ha effetto.

Se i grattacieli rimangono indenni, prima o poi l'aereo perdendo quota, li urterà esplodendo.

```
100 REM***      B O M B E R         ***
110 REM***PROGRAMMA PER COMMODORE 64***
120 REM***           DI             ***
130 REM*** R.LEGNANI & A.SIMONELLI   ***
140 REM*** VIA CASTELVETRO,9-MILANO ***
150 REM***    TELEFONO 02/3496497    ***
151 :
155 REM PER LANCIARE LA BOMBA
156 REM PREMERE IL TASTO "Z"
157 :
160 PRINT"[illegible]ATTENDERE":PRINTCHR$(8)
170 XX$="ERRORE NEI DATA": XX=0
180 V=53248:POKE2042,160:FORN=0TO62:READQ:XX=XX+Q
190 POKE10240+N,Q:NEXT:AA=2
200 IF XX<> 768 THEN PRINT"
210 XX=0:POKE2043,161:FORN=0TO62:READQ:XX=XX+Q
220 POKE10304+N,Q:NEXT
230 IFXX<>671THENPRINTXX$:STOP
240 XX=0:POKE2044,162:FORN=0TO62:READQ:XX=XX+Q
250 POKE10368+N,Q:NEXT
260 IFXX<>2426THENPRINTXX$:STOP
270 XX=0:POKE2045,163:FORN=0TO62:READQ:XX=XX+Q
280 POKE10432+N,Q:NEXT
290 IF XX<>5373THENPRINTXX$:STOP
300 POKEV+42,1:POKEV+43,1:POKEV+44,1
310 PRINT"[illegible]":X=16:Y=64:POKEV+4,X:POKEV+5,Y:POKE53281,4:POKEV+16,0
320 FORC=55296TO56295:POKEC,0:NEXT
330 IFOO<>6THENOO=OO+1
340 FORA=1984TO2022:D=INT(RND(1)*7)+OO:FORW=0TOD:POKEA-W*40,221
350 NEXTW,A:SI=54272:PRINT"[illegible]"
360 FORU=1064TO1103:POKEU,64:NEXT
370 PRINT"[illegible] PUNTI   "RRTAB(27)"AEREI"AA
380 RESTORE:POKEV+21,4
390 XX=0:FORD=0TO251:READH:XX=XX+H:NEXT
400 IFXX<>9238THENPRINTXX$:STOP
410 XX=0:FORK=0TO24:READH:XX=XX+H:POKESI+K,H:NEXT
420 IFXX<>856THENPRINTXX$:STOP
430 POKESI+4,65:POKESI+11,65:POKESI+18,65
440 L7=41
450 X=X+4
460 QW=1024+(INT(X-16)/8)+(((Y-40)/8)*40):IFPEEK(QW)=221THEN880
470 IFQW=2023THENAA=AA+1:GOTO310
480 IFX>255THENPOKEV+16,4:POKEV+4,X-255:GOTO 500
490 POKEV+4,X:GOTO510
500 IFX>327THENPOKEV+16,0:Y=Y+8:X=16:POKEV+5,Y:POKEV+4,X
```

```
510 IFK=1THEN540
520 A=PEEK(203):IFA=12THENK=1:S=QW+1:POKES,46
530 GOTO450
540 D4=D4+1:IFD4=3THEND4=0:GOTO560
550 GOTO450
560 IFPEEK(S+1)=221ANDPEEK(S+40)<>221THENK=0:POKES,32:GOTO450
570 IFPEEK(S+40)=221ANDPEEK(S-40)<>221THENM=M+1:RR=RR+1:L7=40
580 POKES,32:S=S+L7:POKES,46
590 IFM=4ORS>1983THENK=0:M=0:L7=41:POKES,32:PRINT""TAB(8);RR
600 GOTO450
610 POKE 650,0
620 REM SPRITE AEREO
630 DATA 0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
640 DATA 0,0,0,0,0,0,0,0,0,0,0,0,0,1,128,0
650 DATA1,254,0,1,255,128
660 DATA0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
670 REM SPRITE SCOPPIO I
680 DATA0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
690 DATA 0,10,0,0,161,0,0,10,0,1,80,128
700 DATA 0,36,0,0,139,0,0,32,0,0,74,0
710 DATA0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
720 REM SPRITE SCOPPIO II
730 DATA 0,0,0,0,0,0,0,0,0,36,128,0
740 DATA 147,32,2,137,80,1,82,128,4,10,40
750 DATA 4,161,0,2,10,88,9,80,160,4,36
760 DATA 36,8,139,64,10,32,176,5,74,0,0,161,64
770 DATA2,21,32,1,84,128,0,8,0,0,0,0,0,0,0
780 REM SPRITE SCOPPIO III
790 DATA 0,165,0,5,90,208,17,74,168,42,36
800 DATA 200,36,147,42,154,137,84,69,82,129
810 DATA 180,10,42,76,161,4,146,10,89,105
820 DATA 80,162,20,36,44,200,139,66,42,32,181
830 DATA 213,74,18,32,161,76,162,21,34,85
840 DATA 84,145,13,41,44,18,148,136,4,74,80
850 REM DATA PER SUONO
860 DATA 9,2,0,3,0,0,240,12,2,0,4,0,0
870 DATA 192,16,2,0,6,0,0,64,0,30,243,31
880 POKESI+4,0:POKESI+11,0:POKESI+18,0
890 POKEV+16,PEEK(V+16)AND4:F=X
900 IFX>255THENF=F-255:POKEV+16,PEEK(V+16)OR56
910 POKEV+6,F-8:POKEV+7,Y:POKEV+8,F-8
920 POKEV+9,Y:POKEV+10,F-8:POKEV+11,Y
930 AA=AA-1:POKEV+21,8:FORG=1TO400:NEXT
940 POKEV+21,16:FORG=1TO400:NEXT
950 POKEV+21,32:FORG=1TO400:NEXT:POKEV+21,0
960 IFAA<0THEN980
970 Y=Y-32:POKEV+5,Y:POKES,32:PRINT""TAB(27)"AEREI"AA:GOTO380
980 PRINT"PUNTI";RR
990 PRINT"PER UN'ALTRA PARTITA PREMI UN TASTO"
1000 IFPEEK(203)=64THEN1000
1010 AA=2:RR=0:OO=0:GOTO310
READY.
```

L'INFORMATICA A SCUOLA: TELEMATICA, ROBOTICA, BUROTICA

La sostanziale differenza tra la cultura antica e quella moderna sta nel modo di porsi di fronte all'oggetto. Il centro delle attività intellettuali, dall'antichità fino a Galileo, risiedeva nella narrazione, cioè nella descrizione argomentata dei fenomeni. Oggi risiede nelle informazioni tratte scientificamente dalla realtà. In questo quadro l'informazione elaborata, cioè l'informatica, è destinata ad assumere un ruolo sempre più significativo.

L'elettronica è una scienza assai giovane e ci dà l'impressione di essere di fronte ad un iceberg, del quale conosciamo la vetta affiorante ma non la montagna sommersa. Di certo possiamo dire che l'elaboratore e la microelettronica saranno i protagonisti e gli elementi motore dei cambiamenti che investiranno tutte le attività umane, pubbliche e private. Per ora si individuano già almeno tre discipline che trovano nome per aggregazioni d'uso: telematica, burotica e robotica. I cambiamenti non saranno automatici ed indolori, perchè hanno come condizione che la società si evolva culturalmente nella direzione della conoscenza generalizzata dell'informatica. A questo sforzo di aggiornamento saremo chiamati tutti, per esigenze connesse all'efficienza ed alla produttività, ma in cambio ci darà una nuova sicurezza e consapevolezza di decisione ad ogni livello ed in ogni direzione. Prima vinceremo le nostre resistenze alle nuove tecnologie, dovute ad ingiustificate diffidenze o disinformazione, prima saremo consciamente presenti nella realtà sociale e lavorativa di domani. Ogni cultura ha le sue terminologie. Non conoscerne il significato vuol dire esserne esclusi; senza paradosso si può affermare che fra un ventennio chi non conoscerà l'informatica sarà un escluso, come un analfabeta nel mondo d'oggi.

TELEMATICA

Il nome nasce come contraddizione dei termini telecomunicazioni ed informatica. Il suo scopo è quello di fornire dei servizi canalizzati su reti di telecomunicazione, al fine di rendere disponibili le informazioni contenute nelle banche dati, che volgarmente potremmo definire archivi elettronici. Il progresso si misurerà dalla velocità di trasmissione dell'informazione, dalla capacità di immagazzinarla, elaborarla e ritrasmetterla.

Telefono, il terminale più diffuso nel mondo (500/600 milioni di apparecchi), il Televisore e Personal Computer, sono gli strumenti attraverso cui passerà il cambiamento. Già oggi sono in funzione TELETEXT e VIDEOTEXT. Le informazioni, con il Teletext, sono inviate utilizzando un canale televisivo e l'alternativa che ha l'utente è di vederle tutte o di scartarle completamente, con il Videotext, l'informazione viaggia su un cavo telefonico e l'utente può interloquire con la banca dati mediante un telecomando, scegliendo via via quello che desidera ricevere e scartando ciò che non gli interessa.

I fornitori di informazioni, in inglese «Information Provider», sono banche, borse, università, enti pubblici ed organizzazioni private, i cui servizi possono essere gratuiti e/o a pagamento.

ROBOTICA

La parola robot venne coniata nel 1917 dallo scrittore cecoslovacco K. Capek, che la usò nel racconto breve «R.U.R.». Deriva dal ceco «ROBOTA» e significa lavoro duro, servile e/o coatto, quello dei servi della gleba. L'idea di creare un essere del tutto simile all'uomo, ma ad esso asservito, che svolgesse i lavori più ingrati, è vecchio quanto l'uomo stesso.

La letteratura si è più volte appropriata di questo mito, ma in una chiave del tutto particolare: la creatura si rivolta al suo creatore che solo alla fine riesce ad averne ragione distruggendola. L'esempio più famoso è Frankenstein. Purtroppo ancora oggi grosse produzioni cinema-

COMPUTER QUESTO MESE È QUESTO

Computer

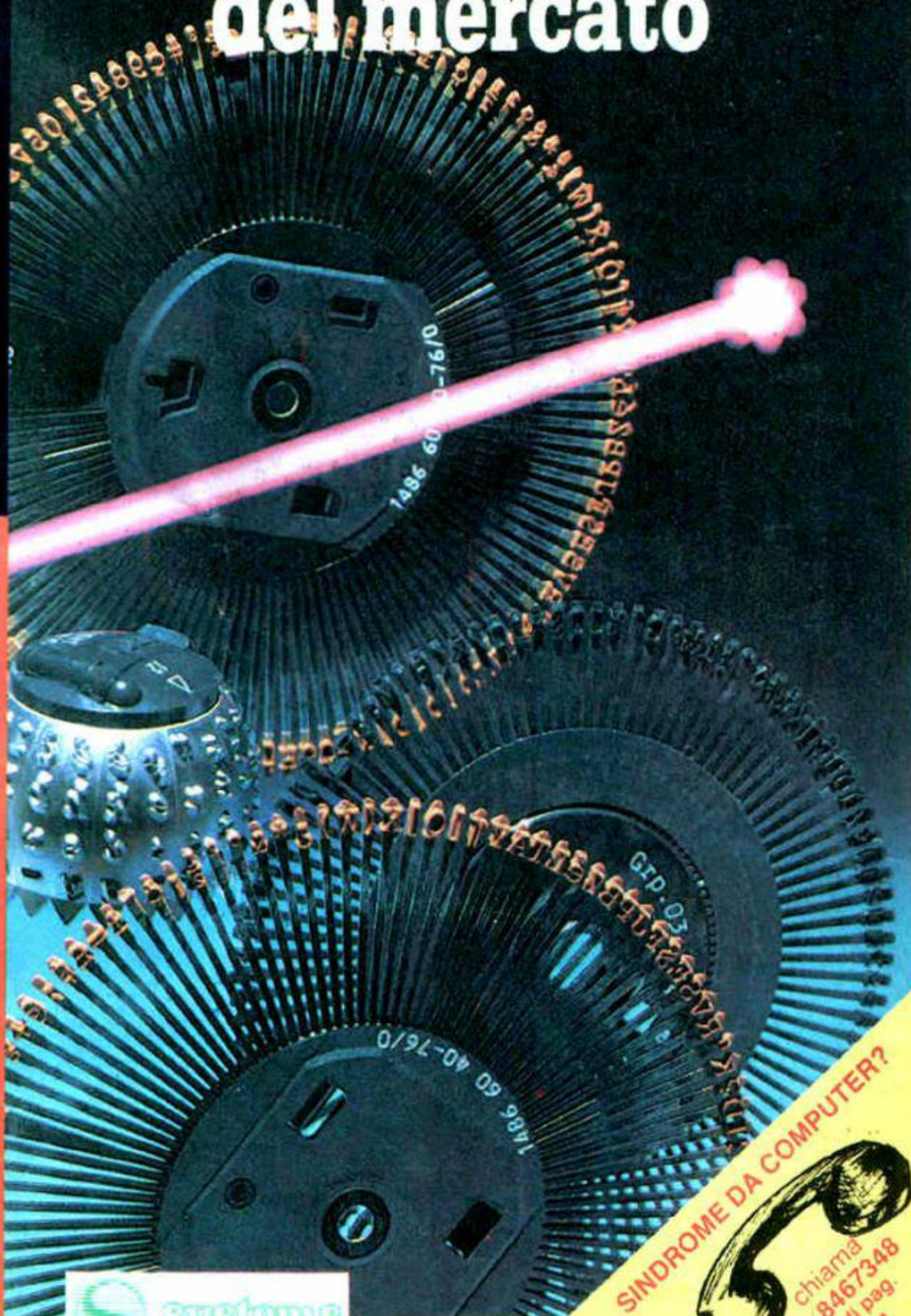
N.69 - lire 3000 il "newsmagazine" dell'informatica

Tutte le stampanti del mercato

Cosa vedere al prossimo Smau

Data base: come sceglierli

Ed ecco il transputer

Benchmark: il People Olympia un anno dopo

systems

SINDROME DA COMPUTER? chiama 02/8467348 vedi pag. 41

Computer Anno VIII n. 69 25 agosto-25 settembre 1984 - Sped. Abb. Post. Gr. III/70 - CH - Distr. MePe

tografiche si basano su questo equivoco, insinuando ingiustificati timori e danneggiando un armonioso sviluppo dell'informatica. Nel 1950 Isac Asimov nel celeberrimo romanzo «Io, Robot», coniò le tre leggi della robotica, nelle quali è riassunta la moderna concezione di robot:

- Un robot non può recar danno ad un essere umano, nè può permettere che, a causa del proprio mancato intervento, un essere umano riceva danni.
- Un robot deve obbedire agli ordini impartiti dagli esseri umani, purchè tali ordini non contravvengano alla prima legge.
- Un robot deve proteggere la propria esistenza, purchè questa autodifesa non contrasti con la prima e la seconda.

Oggi il campo di impiego dei robot è soprattutto industriale. Grande sviluppo stanno per avere i robot nelle industrie con catene di montaggio, in particolare quelle automobilistiche (Robogate), ma anche in molti altri settori dove svolgono compiti per l'uomo pericolosi, impossibili, faticosi o ripetitivamente noiosi. I robot industriali possono essere usati in più mansioni, e ciò presuppone che accanto ad una struttura meccanica esista una struttura elettronica, basata su un microprocessore ed una memoria.

BUROTICA

E' un termine di derivazione francese: «bureau» cioè ufficio e «informatique» si sono contratti in «bureautique», che nell'uso corrente italiano è diventato «burotica». La burotica si occupa delle tecniche e delle tecnologie relative all'automazione dell'ufficio.

La storia dell'organizzazione del lavoro di ufficio mostra i vari periodi cambiamenti, soprattutto ad opera di innovazioni tecnologiche, ma per la relativa eseguità sociale degli impiegati rispetto agli operai, l'aspetto organizzativo è sempre stato subordinato al rapporto personale e all'ordine gerarchico, nonostante che nello stesso periodo in fabbrica l'organizzazione del lavoro fosse molto cambiata. Solo la grande crescita del numero degli impiegati, fino a diventare maggiore di quello degli operai, ha posto problemi organizzativi. L'introduzione del computer, che ha portato l'esigenza di procedure e metodi standardizzati è stata l'elemento catalizzatore di questo processo.

L'organizzazione funzionale sta avendo il sopravvento sull'organizzazione gerarchica; ora l'azienda è sempre più un sistema informativo. Prendiamo ad esempio il passaggio dalla configurazione accentrata dell'elaboratore alle configu-

Computer questo mese é questo...

E QUESTO ....

ALBERTO CULTRERA

1994

IL MONDO DEI COMPUTER AMICI

COMPUTER

razioni decentrate presso i singoli uffici, ciò che si definisce correntemente l'informatica distribuita, a questa corrisponde un decentramento di competenze e responsabilità.

Il Word-Processing (che permette l'elaborazione dei testi e la loro archiviazione) e la possibilità di digitalizzare la voce umana inseriti nel sistema informativo dell'ufficio, tramite lo sviluppo della telematica, ci faranno giungere ad un sistema informativo a livello planetario. E' chiaro che in breve tempo, nei paesi industriali molti potranno o dovranno accedere ad un elaboratore personale, quindi il grande sforzo da compiere sarà quello della formazione informatica di massa.

Finalmente una Scuola per Diventare Progettisti di Videogiochi

Col ritorno dalle vacanze estive una grossa novità attende gli appassionati di home-computer e videogiochi: sarà inaugurata la Video-design School, la scuola che è nata per insegnarvi a programmare il vostro videogioco. I corsi saranno articolati su tre livelli, per principianti, esperti, specialisti ed alla fine verrà rilasciato un attestato di idoneità (a chi sarà promosso!).

L'esame sarà costituito dalla progettazione e realizzazione di un gioco, che verrà discusso in un colloquio. I candidati potranno scegliere liberamente se cimentarsi su temi di intrattenimento o didattici.

Per informazioni, telefonare al 02/6899912 - 603868.

INVIARE TUTTA LA PAGINA ANCHE SE SI UTILIZZA UNA SOLA SCHEDA

Nome ______________ Cognome ______________

Via ______________ N° ________ CAP. ________ Città ______________

Telefono ______________ Orario ______________

Registrate il mio abbonamento annuale a Commodore Computer Club:

☐ Ho versato oggi stesso il canone di Lire 22.000 a mezzo c/c postale n° 31532203 intestato a: Commodore Computer Club - V.le Famagosta, 75 - 20142 Milano

Il mio computer è: ☐ un VIC 20, ☐ un C 64, ☐ altro (specificare) ______________

☐ ho, ☐ non ho, ☐ la stampante, ma voglio ☐ comprarla

☐ ho, ☐ non ho, ☐ il drive 1541, ma voglio ☐ comprarlo.

Registratemi fra i collaboratori regolari di Commodore Computer Club.

A titolo di prova vi invio un articolo e la cassetta col programma " ______________ "

di cui vi garanatisco l'assoluta originalità autorizzandovene la pubblicazione.

Per il compenso ☐ telefonatemi al ________ ☐ Scrivetemi all'indirizzo sottoindicato.

Servizio Domanda/Risposta ______________

Questo mese i programmi / articoli li ho giudicati così:

Giudizio d'utente (mettere una x nella casella)

TITOLO	OTTIMO	BUONO	SUFFICIENTE	MEDIOCRE

INVIARE IN BUSTA CHIUSA E AFFRANCANDO SECONDO LE TARIFFE VIGENTI A:

COMMODORE COMPUTER CLUB

V.le Famagosta, 75
20142 Milano

FINALMENTE.
LA TAVOLETTA GRAFICA A PIENE PRESTAZIONI AD UN PREZZO ACCESSIBILE A TUTTI

Koala
Disponibile per Apple][+ e //e
Atari 400 e 800, Commodore
ed IBM P.C.

La tavoletta grafica KOALA è la più simpatica innovazione nel campo dei personal computers. Con KOALA, controllate il vostro computer con un dito. Più veloce di un paddle, più versatile di un joystick e più semplice di una tastiera.
La tavoletta grafica KOALA è compatibile con la maggior parte di software esistente e viene fornita completa del suo programma grafico "Micro Illustrator". KOALA-PAD è il miglior modo per creare immagini ad alta risoluzione con il vostro computer.

TELAY INTERNATIONAL s.r.l.
COMPUTER GRAPHICS DIVISION
MILANO: Via L. da Vinci, 43 - 20090 Trezzano S/N
Tel. 02/4455741/2/3/4/5 - Tlx: TELINT I 312827
ROMA: Via Salaria, 1319 - 00138 Roma
Tel. 06/6917058-6919312 - Tlx: TINTRO I 614381